Universidad Central "Marta Abreu" de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Telecomunicaciones y Electrónica

DISEÑO DE UN SISTEMA DE CONTROL SUPERVISORIO, BASADO EN WEB, PARA DISPOSITIVOS DE CONTROL GESTIONABLES A TRAVÉS DE HTTP.

Tesis presentada en opción al Título Académico de
Master en Telemática

Maestría de Telemática

Autor: Ing. Nono Francisco Carballo Escalona

Tutor: Dr. Pablo Pérez Ajo

2006

A mi esposa María Angeles y a mi hija Ángela Gabriela

RESUMEN

El éxito alcanzado por Internet en la última década corrobora la consolidación del protocolo TCP/IP como estándar en los servicios y aplicaciones. En los últimos tiempos se ha visto como este protocolo se usa no solamente en los entornos computacionales, sino que ha abarcado otro tipo de dispositivos como los dispositivos de control. Los productores de este tipo de dispositivos han comenzado a incluir el protocolo TCP/IP como vía de comunicación, a utilizarse en tareas como el monitoreo, la configuración, entre otras. Los sistemas de control supervisorio, por ende, también han comenzado a incorporar esta vía de comunicación con los dispositivos.

Un éxito similar ha tenido el protocolo HTTP, el cual, de la misma forma que TCP/IP, está siendo incorporado en estos tipos de dispositivos para soportar las tareas antes mencionadas. HTTP en la actualidad no solo se utiliza como soporte para la ejecución de aplicaciones Web, sino también para el desarrollo de las mismas; aspecto este en el que se aprecia una notable evolución, y en la actualidad se encuentran disponibles diferentes tecnologías soportadas por una variedad de lenguajes y plataformas.

El presente trabajo aborda el diseño de un sistema de control supervisorio, basado en Web, para dispositivos de control gestionables a través de HTTP. Se establecen las especificaciones técnicas del sistema, se diseña y se define la arquitectura del mismo utilizando la metodología Rational Unified Process, Del mismo modo de selecciona, entre las técnicas de desarrollo de aplicaciones Web mas utilizadas, la más conveniente para la etapa de desarrollo.

TABLA DE CONTENIDO

LISTADO DE FIGURAS

LISTADO DE TABLAS

INTRODUCCIÓN

La utilización del protocolo TCP/IP ha marcado un hito en el uso de las redes de computadoras. Su implementación sobre una gran variedad de protocolos de nivel físico ha propiciado su amplia presencia no solamente en computadoras personales sino también en una amplia gama de dispositivos que ofrecen la posibilidad de conectarse a redes de área local para su gestión.

Entre esos dispositivos se encuentran los que son diseñados para el control de diferentes procesos., los denominados PLC y RTU. Estos dispositivos, que tradicionalmente disponen de conexiones RS232 y/o RS485 para su comunicación exterior, en la actualidad incorporan interfaces de la familia Ethernet y otras tecnologías de redes y se observa una tendencia a la incorporación de capacidades de gestión a través del protocolo HTTP.

Todo este progreso hace posible que la información adquirida por estos dispositivos de control pueda ser accedida a través de la red de área local utilizando los protocolos TCP/IP y HTTP, lo que trae como consecuencia una evolución en los sistemas de supervisión, que anteriormente se comunicaban con los dispositivos de control a través de sus interfaces tradicionales, y que en la actualidad incorporan estas vías de comunicación.

Hoy en día los productores de sistemas de control supervisorio comerciales, ante la creciente aparición en el mercado de dispositivos de control gestionables a través de HTTP, han comenzado a incorporar esta forma de comunicación a sus sistemas.

El hecho que los dispositivos sean gestionables a través de HTTP abre grandes posibilidades a la estandarización en la operación con los mismos, a diferencia de lo que ha ocurrido hasta ahora, en que los protocolos propietarios y productos de software hacen de éste un mercado bastante complicado. HTTP es uno de los protocolos de aplicación más utilizados en Internet y su desarrollo está dirigido por organismos internacionales como el World Wide Web Consortium (W3C), que se encarga de aprobar y promover el uso de estándares relacionados con el mismo.

En la actualidad HTTP y el servicio de Internet asociado a él WWW no se limitan a servir como soporte de sistemas de información, sino que han abarcado otros campos. Uno de ellos es la

gestión, como se ha mencionado anteriormente y otro es el desarrollo y ejecución de aplicaciones.

El protocolo HTTP, debido a su simplicidad, su accesibilidad, gran cantidad de productos que lo implementan, y a pesar de su característica de no mantener estado entre sesiones; está siendo ampliamente utilizado no solamente en el terreno de la ejecución de aplicaciones, sino también como soporte de herramientas de desarrollo de aplicaciones (que tradicionalmente eran aplicaciones de escritorio). Los sistemas de control supervisorio pueden encontrar en el servicio WWW una forma de aumentar su accesibilidad al poder ser alcanzados desde cualquier punto de la red, simplificar los procesos de instalación, actualización y despliegue, así como la disminución de los requerimientos de ejecución en el cliente.

El presente trabajo aborda el diseño de un sistema de control supervisorio para dispositivos de control gestionables a través del protocolo HTTP, para lo cual debe dar respuestas a las siguientes interrogantes:

- ✓ ¿Cuál debe ser la arquitectura de un sistema de control supervisorio para dispositivos gestionables por HTTP?
- ✓ ¿Cuáles deben ser sus requisitos funcionales?
- ✓ ¿Cuál debe se ser su modo de operación?

OBJETIVOS

El trabajo pretende elaborar toda la información necesaria para la elaboración, en términos computacionales, del sistema de control supervisorio, lo cual se logrará a través de los siguientes objetivos particulares:

- ✓ Elaboración de las especificaciones técnicas del sistema supervisor
- ✓ Diseño del sistema utilizando una metodología de desarrollo de software.
- ✓ Diseño de la arquitectura del sistema supervisor.

RESULTADOS ESPERADOS

Para cumplir los objetivos propuestos, el presente trabajo se sustentará en la utilización de la metodología de desarrollo de software Rational Unified Process, la cual, en sus diferentes etapas de trabajo, produce salidas denominadas artefactos. En este trabajo se obtendrán los artefactos correspondientes a las etapas de Gestión de Requisitos, Análisis y Diseño.

EVALUACIÓN DEL IMPACTO

Los sistemas de control supervisorio son productos de software de alto valor agregado, de ahí su elevado precio en el mercado. En la actualidad todos los productos de este tipo disponibles comercialmente están soportados sobre sistemas operativos y tecnologías propietarias, lo que encarece notablemente la ejecución de cualquier proyecto de automatización que los utilice.

Nuestro país se encuentra en este momento inmerso en diversos proyectos que convergen en un objetivo común, la informatización de la sociedad cubana. En este proceso juega un papel fundamental la utilización de tecnologías gratuitas y de código abierto. Si bien en este trabajo no se lleva a cabo la etapa de programación, en la cual se ve una mayor incidencia de estos factores, se recomienda grandemente la utilización de tecnologías de este tipo, en aras de contar con un producto de software extensible, y de bajo costo.
La automatización de las diferentes esferas de la producción y de la vida social también forma parte de la informatización de la sociedad, ya sea una industria, un hotel, un edificio. El producto de software que se cree utilizando los resultados de este trabajo, podrá ser utilizado en cualquier proyecto de automatización que utilice dispositivos gestionables a través de HTTP, al ser un producto elaborado mediante tecnologías gratuitas y de código abierto, contribuirá a la disminución del costo de los proyectos de automatización.

METODOLOGÍA DE TRABAJO

Para la elaboración de este trabajo se realizó una búsqueda bibliográfica acerca de los aspectos que constituyen los fundamentos teóricos del mismo. Así mismo se realizaron búsquedas en Internet de productos de software similares, sus características técnicas, requisitos, licencias de operación, precios.

Se realizó un estudio de las tecnologías más utilizadas en la actualidad en el desarrollo de aplicaciones Web. El estudio se abordó desde el punto de vista comparativo y finalmente se eligió una de ellas para la ejecución de la etapa de programación.

Se entrevistaron a clientes de sistemas de control supervisorio de diferentes tipos de instalaciones, así como a especialistas de proyectos de automatización. Sus criterios fueron de vial importancia en la elaboración de la especificación técnica y en la etapa de Gestión de Requisitos.

ESTRUCTURA DEL TRABAJO

El trabajo está compuesto por tres capítulos:

- ✓ Capítulo I. El protocolo TCP/IP, HTTP y los sistemas de control supervisorio
- ✓ Capítulo II. Tecnologías de desarrollo de aplicaciones Web
- ✓ Capítulo III. Diseño de un sistema de control supervisorio para dispositivos gestionables por HTTP

En el primer capítulo se hace una breve reseña de los elementos fundamentales que intervienen en la elaboración de un producto como el que se diseña. Se realiza un breve repaso acerca del protocolo TCP/IP, el cual constituye el protocolo de transporte sobre el cual se cimienta la aplicación. Se hace un breve estudio también del protocolo HTTP, el cual se utilizará para comunicar la aplicación con los dispositivos de control que se monitorearán con la misma. Finalmente se aborda el tema de los sistemas de control supervisorio de manera general, se mencionan las características de estos sistemas y la arquitectura mas difundida en la literatura. En este capítulo no se pretende analizar a profundidad estos aspectos, debido a la extensión de los mismos, solamente se tocarán aquellos temas que son importantes en el desarrollo de un sistema como el que plantea en este trabajo.

El segundo capítulo aborda las tecnologías de desarrollo de aplicaciones Web. Se parte de las primeras técnicas que se utilizaron para crear contenido dinámico en entorno Web y se tratan las tecnologías más utilizadas en la actualidad para este tipo de aplicaciones. Finalmente, haciendo un análisis de éstas se selecciona la que se utilizará para la fase de implementación de la aplicación.

El tercer capítulo está dedicado al diseño del sistema de control supervisorio utilizando la metodología Rational Unified Process. Inicialmente se realiza un breve análisis acerca de la metodología y su surgimiento. Se analizan además las diferentes fases y flujos de trabajo que las componen, indicando los artefactos resultantes de cada uno de ellos. Posteriormente se detalla el diseño del sistema objeto de estudio en este trabajo. Los artefactos resultantes de las diferentes etapas de trabajo se reflejan en la sección de Anexos.

CAPÍTULO I. EL PROTOCOLO TCP/IP, HTTP Y LOS SISTEMAS DE CONTROL SUPERVISORIO.

El uso de las redes de computadoras tuvo un crecimiento notable a partir del surgimiento de la familia de protocolos TCP/IP y del concepto de interconexión de redes. La implementación de este protocolo sobre una gran variedad de tecnologías de nivel físico ha propiciado su amplia aceptación entre los fabricantes de tecnologías de redes y de sistemas operativos, su implementación sobre la tecnología Ethernet es una de las más populares, a tal extremo que en la actualidad es casi el protocolo utilizado por excelencia en esos entornos.

La arquitectura de la familia de protocolos TCP/IP está compuesta por cuatro niveles. El nivel más inferior se denomina “del host a la red”, le siguen en orden ascendente el nivel de internet, el nivel de host a host y en la parte superior de la arquitectura se encuentra el nivel de procesos. La figura No. 1 muestra la correspondencia de estos niveles con los niveles de la arquitectura del modelo OSI [1].

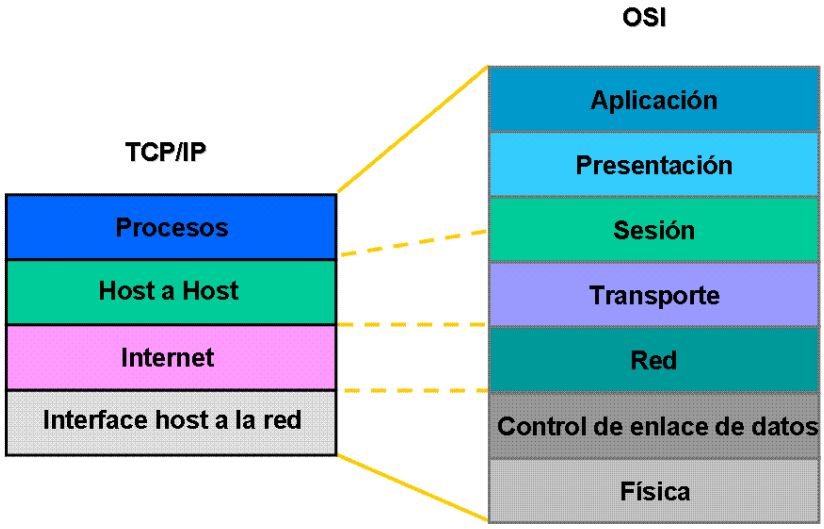


Figura No. 1. Arquitectura por niveles de TCP/IP y del modelo OSI

1.1 LA FAMILIA DE PROTOCOLOS TCP/IP

Esta familia de protocolos, caracterizada por su simplicidad de implementación, está compuesta fundamentalmente por los protocolos TCP (Transmisión Control Protocol), UDP (User Datagram Protocol), IP (Internet Protocol) e ICMP (Internet Control Message Protocol), pero son los protocolos TCP e IP los mas populares y que le han dado el nombre. La figura No. 2 muestra la ubicación de los protocolos mencionados en la arquitectura por niveles.

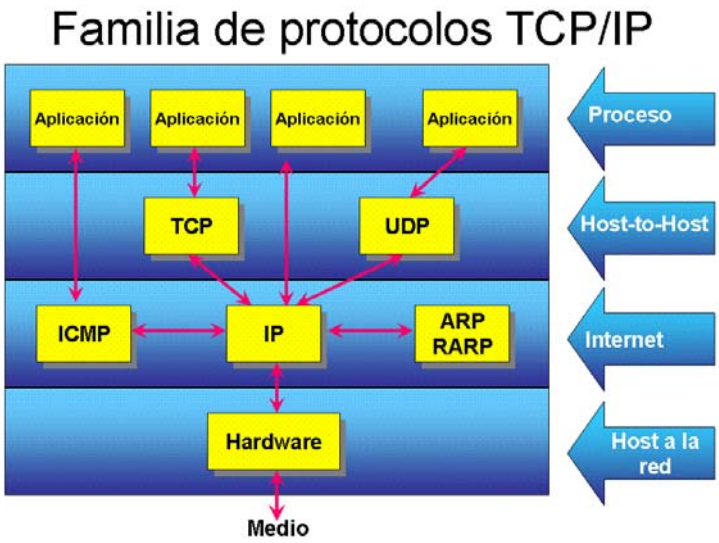


Figura No. 2. Ubicación de los protocolos de TCP/IP en la arquitectura por niveles

1.1.1 El protocolo IP

El protocolo IP, está diseñado para ser usado en sistemas interconectados de redes de comunicación de conmutación de paquetes [2]. El mismo transmite bloques de datos llamados datagramas desde el nodo emisor hasta el nodo de destino, los cuales están identificados por direcciones de longitud fijas, llamadas direcciones IP [1].; es el responsable de llevar a cabo la fragmentación, el reensamblado y el direccionamiento. La versión en uso en la actualidad es la versión IV.

Una red TCP/IP está compuesta por los hosts desde donde se originan y hacia donde se destinan los datagramas y elementos intermedios denominados enrutadores, encargados de determinar, utilizando las direcciones de los host, el camino a seguir por cada datagrama, los cuales son tratados de forma independiente.

Por diseño, el protocolo IP está destinado solamente a suministrar las funciones necesarias que le permitan transmitir un flujo de bits (un datagrama) desde el nodo emisor hasta el destino; por lo que no provee de mecanismos que garanticen una entrega confiable, ordenada, con control de flujo entre las partes que intervienen en la comunicación; del mismo modo, no hay reconocimientos entre los extremos de la comunicación ni entre los elementos intermedios, no hay controles de error del dato (solo una suma de verificación en el encabezado), no hay retransmisiones; de ahí que se dice que IP es un protocolo no orientado a conexión, no confiable y de mejor esfuerzo. Los errores detectados se reportan mediante otro protocolo de la familia, el ICMP [3]

Los módulos de Internet, presentes en cada host conectado a la red y los elementos intermedios, utilizan la dirección que se encuentra en el encabezamiento del datagrama IP para transmitirlo hacia su destino; para lo cual se auxilian de un mecanismo que le permite determinar el camino para esa transmisión, denominado enrutamiento. Puede decirse entonces que el objetivo del protocolo IP es transmitir datagramas entre los módulos de Internet de los host interconectados entre el origen y el destino

Para transmitir un datagrama entre el host de origen y un host de destino, la aplicación que envía el datagrama, ejecutándose en el host de origen, prepara la información y la entrega al módulo de internet local (a través del módulo de host a host), unido a la dirección IP del host de destino, entre otros parámetros. El módulo de internet prepara un encabezado (figura No. 3) y lo añade al datagrama (a este proceso se le conoce como encapsulamiento), también determina una dirección de red para la dirección IP del host de destino. Si el host de destino no se encuentra en la misma red que el host emisor se entregará el datagrama a un elemento intermedio, y en este caso la dirección de red mencionada anteriormente corresponderá al elemento intermedio; en caso contrario se entregará al host de destino directamente. A continuación se envía el datagrama y la dirección de red computada a la interfase de red local, la cual crea un encabezamiento, lo añade al datagrama y lo envía a través de la red local.

El datagrama arriba al host identificado con la dirección de red computada anteriormente, donde la interfase de red local extrae el encabezamiento que le corresponde y lo entrega al módulo de internet, donde se determina si éste ha llegado a su destino o si debe ser enviado a algún otro host. En el primer caso el datagrama es entregado a los módulos superiores para posterior procesamiento. En caso que el datagrama haya arribado a un elemento intermedio, el módulo de Internet determina, basado en la dirección IP del host de destino, la dirección de red del próximo host al que debe ser enviado el datagrama para su distribución, entrega nuevamente el datagrama a la interfase de red local la cual se encargará de transmitirlo por la red nuevamente [2].

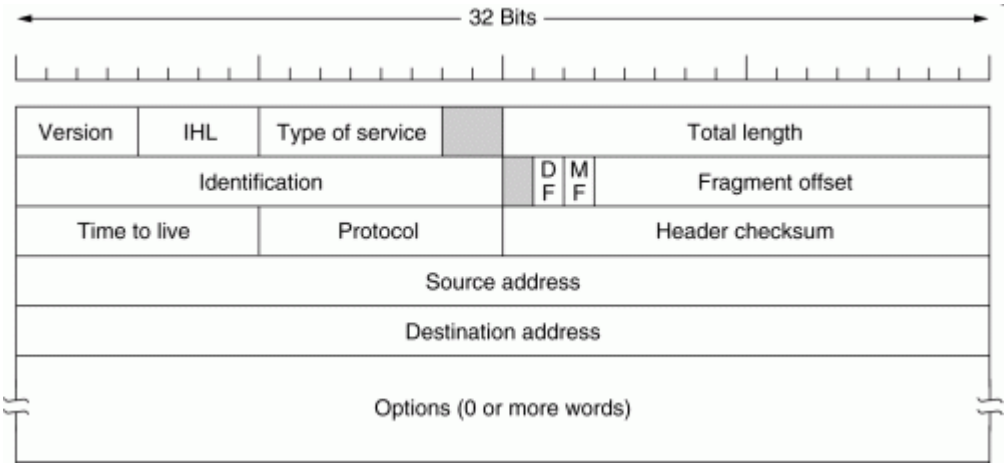


Figura No. 3. Encabezamiento del protocolo IP [1].

1.1.1.1 DIRECCIONAMIENTO

Como se mencionó anteriormente, el direccionamiento es una de las responsabilidades fundamentales del protocolo. Éste hace distinción entre nombres, direcciones y rutas. Un nombre representa el host al que se quiere alcanzar, la dirección indica dónde se encuentra el host, en que red, y la ruta expresa cómo alcanzar a ese host [2]. IP trabaja con direcciones, delegando a las aplicaciones de los niveles superiores la responsabilidad de encontrar una dirección IP asociada a un nombre de host.

Una dirección representa de forma única una conexión de un host a la red. Un host puede tener múltiples conexiones a la red a través de varias interfases de red (tal es el caso de los multihomed hosts), pero a cada conexión le corresponderá una dirección IP diferente. Las direcciones IP son números de 32 bits, representados en una notación de cuatro octetos separados por un punto. Cada dirección está dividida en dos partes, una que identifica la red a la que pertenece y la otra que identifica a una conexión en particular dentro de la red.

Las direcciones IP están divididas en cuatro clases, A, B, C y D, tomando como criterio para esta división el valor de los cuatro primeros bytes del primer octeto. Existe además una clase denominada E que se reserva para uso futuro y cuyas direcciones no deben ser asignadas a ninguna conexión. La figura 4 ilustra la clasificación de las direcciones IP en clases y el rango de direcciones asociadas a cada clase.

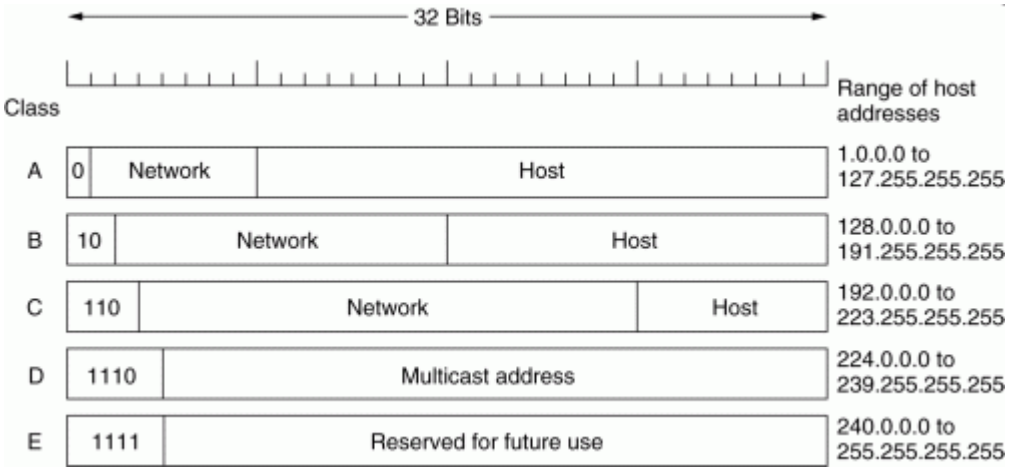


Figura No. 4. Clasificación de las direcciones IP [1].

Existen direcciones que tienen un significado especial para el protocolo y que no deben ser asignadas a ninguna conexión, éstas están reflejadas en la figura 5.

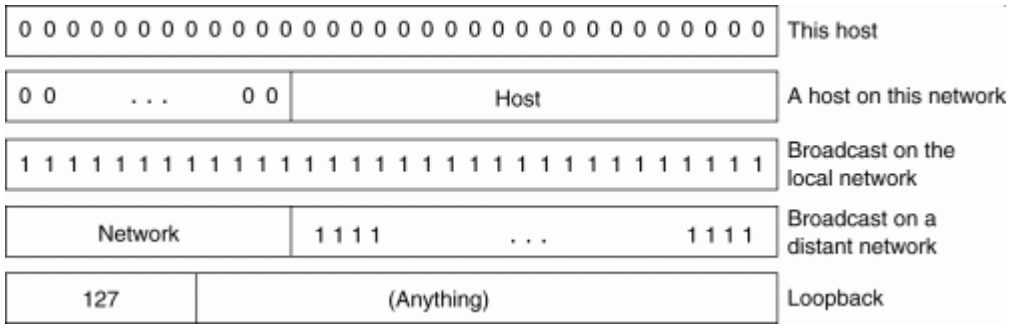


Figura No. 5. Direcciones IP especiales [1].

Cada clase de direcciones tiene una cantidad de bits reservados para la parte de host, lo que determina la capacidad de direccionamiento de la misma, ésta se muestra en la tabla No. 1.

Tabla No. 1 Capacidad de direccionamiento de host en cada clase de direcciones IP

Clase	Bits en la parte de red	Bits en la parte de host	Cantidad de host a direccionar
A	8	24	16,777,214
B	16	16	65,534
C	24	8	254

Para una mejor administración del espacio de direcciones se introduce el concepto de subredes (subneting). Mediante esta técnica se divide un espacio de direcciones dado en múltiples subredes para su uso interno dentro de la organización a la que ha sido asignada. Varios son

los criterios que pueden utilizarse para tomar la decisión de subdividir una red, entre los más frecuentes se encuentran:

- ✓ Separación geográfica de las subredes
- ✓ Separación funcional de las subredes
- ✓ Control de tráfico
- ✓ Seguridad

La cantidad de hosts y subredes que es posible asignar en una subred está dada por la expresión

$$\mathbf{2^n - 2}, \quad (1)$$

Donde n es la cantidad de bits empleados para direccionar hosts o subredes.

Para el funcionamiento de este mecanismo es necesario introducir el concepto de máscara de subred. La máscara de subred es un número de 32 bits que se representa mediante cuatro octetos separados por un punto (al igual que la dirección IP), con la particularidad que los bits correspondientes a la parte de red y subred se encuentran siempre con el valor 1 y los bits correspondientes a la parte de host siempre tendrán el valor 0. Todas las direcciones IP pertenecientes a una misma subred comparten el mismo valor de mascara de subred.

La máscara natural es la máscara que se utiliza en una subred clase A, B o C sin subdividir, estas máscaras se listan en la tabla No. 2

Tabla No. 2. Máscaras naturales para las clases A, B y C

Clase	Máscara
A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

Para ejemplificar el mecanismo de subred tomemos la dirección de clase C 172.16.10.0 con máscara 255.255.255.0. Esta es una dirección clase C que utiliza la máscara natural, o sea, no ha sido subdividida. Con 8 bits disponibles para direccionar host podemos asignar, según (1), 254 direcciones diferentes. Si se utilizamos 5 bits para subdividir esta dirección, la máscara correspondiente sería 255.255.255.248, y con esta máscara se disponen de 5 bits para subredes y 3 bits para hosts, lo cual permite direccionar 30 subredes de 6 hosts cada una. La tabla No. 3 muestra los valores de las máscaras y la capacidad de direccionamiento resultante al subdividir una clase C.

Tabla No. 3 Subredes de una clase C

# de bits de subred	Máscara	No. de subredes	No. De hosts
2	255.255.255.192	2	62
3	255.255.255.224	6	30
4	255.255.255.240	14	14
5	255.255.255.248	3	6
6	255.255.255.252	62	2

La máscara de subred juega un papel fundamental en el proceso de transmisión de un datagrama. En el proceso de transmisión de un datagrama entre el host emisor al host de destino se debe determinar, conocida la dirección IP del host de destino, si ambos se encuentran en la misma subred o no. En el primer caso la entrega se realizará directamente al host de destino, en caso contrario se enviará el datagrama a un elemento intermedio a cuyo cargo queda la retransmisión del mismo.

Para determinar si el host emisor y destino se encuentran en la misma subred es necesario conocer la dirección IP y máscara de subred de host emisor y la dirección IP del host de destino. Se aplica una operación AND a nivel de bits entre la máscara de subred y la dirección IP del host emisor y otra operación AND entre la mascara de subred del host emisor y la dirección IP del host de destino si el resultado es el mismo ambos hosts se encuentran en la misma subred, en caso contrario se encuentran en subredes diferentes. El algoritmo se muestra en la figura No. 6.

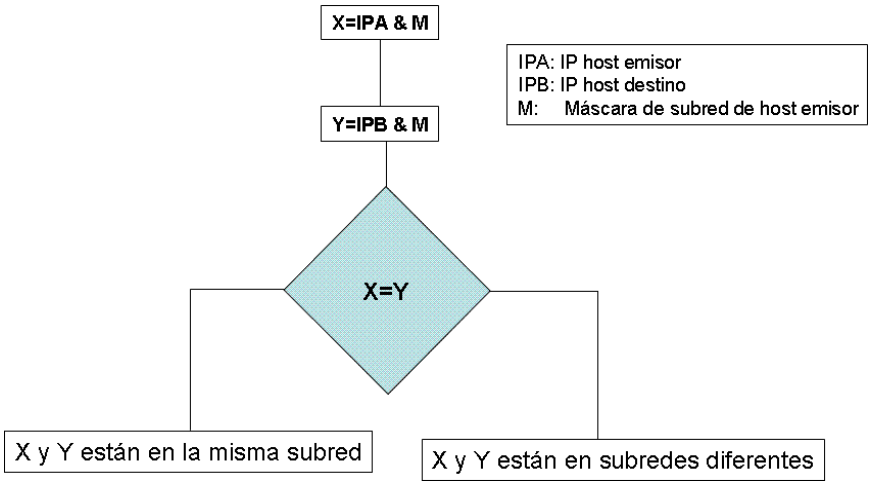


Figura No. 6. Algoritmo para determinar si dos host se encuentran en la misma subred

1.1.1.2 FRAGMENTACIÓN Y REENSAMBLADO

La fragmentación es el proceso que ocurre en los elementos intermedios de una red TCP/IP y consiste en la división de un datagrama en datagramas más pequeños debido a las limitaciones que impone el nivel físico en cuanto al tamaño máximo del paquete a enviar, conocido como MTU (Maximum Transfer Unit).

Cuando el elemento intermedio detecta la necesidad de fragmentar un datagrama, el mismo puede hacerlo en un número arbitrario de éstos y a cada fragmento añade un encabezado idéntico al del datagrama original, pero con los campos relacionados con la fragmentación, actualizados. Los fragmentos serán reensamblados solamente en el host de destino; o sea, una vez que un datagrama ha sido fragmentado continuará de esa forma aunque todos los fragmentos sigan el mismo recorrido y pasen a una red con MTU mayor que el MTU de la red de donde provenían. Un datagrama puede ser marcado de forma tal que impida ser fragmentado, situando una marca de "*no fragmentación*" en el mismo. Si un elemento intermedio necesita fragmentar un datagrama marcando de esa forma, en vez de realizar la operación de fragmentación descartará el datagrama.

El host de destino para reensamblar un datagrama consulta la información ofrecida por el encabezado del datagrama (ver figura No. 3), mediante la cual puede conocer si es datagrama es parte o no de un datagrama fragmentado y si lo es a que porción del datagrama original corresponde, así como si es el primer o último fragmento del datagrama original.

1.1.2 EL PROTOCOLO TCP

El protocolo TCP ha sido diseñado para ser usado como un protocolo altamente confiable de extremo a extremo, en una red de comunicación de conmutación de paquetes y sistemas interconectados por ésta [4], se ubica en la capa de host a host en la arquitectura TCP/IP (Ver figura No. 2).

Este es un protocolo orientado a conexión que ofrece mecanismos para la comunicación confiable entre dos aplicaciones ejecutándose en hosts conectados a dos redes distintas desde el punto de vista físico, pero interconectados a través de una red de conmutación de paquetes que ofrece servicios no confiables en los niveles inferiores.

Para lograr sus objetivos TCP debe proveer mecanismos eficientes para la transferencia de datos, la confiabilidad, el control de flujo, la multiplexación, el manejo de conexiones y la precedencia y seguridad.

1.1.2.1 TRANSFERENCIA DE DATOS

El protocolo TCP es capaz de transmitir un flujo de continuo de octetos en cada dirección empaquetando un conjunto de esos octetos en segmentos para su transmisión a través del módulo de Internet. TCP decide cuando bloquear y reanudar la transferencia de acuerdo a las características y el estado de la conexión.

Cuando una aplicación entrega datos a TCP, éste puede enviarlos de inmediato o colocarlos en una memoria intermedia (conocida en la literatura en ingles como *buffer*) para enviar más cantidad de información en un solo proceso. Si la aplicación requiere que sus datos sean enviados de inmediato sin ser colocados en el buffer debe utilizar la función PUSH suministrada por la implementación del protocolo, que causa la activación del bit PSH en el encabezado del protocolo (ver figura 7).

1.1.2.2 CONFIABILIDAD

TCP es capaz de recuperarse cuando el dato ha sido dañado, perdido, duplicado o recibido en un orden distinto al que fue enviado. Para esto asigna un número de secuencia a cada octeto enviado y espera un acuse de recibo positivo (ACK) por parte del host de destino.

Al transmitir un octeto se inicia un temporizador de retransmisiones. Si el acuse de recibo positivo llega antes de que expire el temporizador éste se detiene, en caso contrario se retransmite el octeto y se reinicia el temporizador nuevamente [1].

En el host de destino los números de secuencias se utilizan para ordenar los diferentes segmentos y para eliminar los duplicados. Cada segmento trae consigo un valor correspondiente a una suma de verificación, la cual es recalculada en el host de destino para descartar los segmentos dañados. El host de destino puede solicitar al host emisor la retransmisión de un segmento utilizando un acuse de recibo negativo con el número de secuencia del segmento dañado.

1.1.2.3 Control de flujo

TCP ofrece un mecanismo para que el host de destino controle la cantidad de datos que envía el host emisor conocido como de ventana deslizante. La ventana indica el número de octetos que el host emisor puede transmitir sin que haya recibido acuse de recibo del último enviado.

1.1.2.4 Multiplexación

Para permitir que múltiples procesos dentro de un mismo host utilicen las facilidades del protocolo TCP se introduce en concepto de *socket*. Un socket consiste en un par formado por la dirección IP del host y un número d 16 bits denominado *puerto*. Para utilizar el servicio del protocolo debe establecerse una conexión entre un socket en el host emisor y un socket en el host destino. Ambos sockets identifican de forma única cada conexión y pueden ser utilizados simultáneamente en múltiples conexiones, es decir, dos conexiones pueden utilizar el mismo socket en el host de destino.

La asignación de puertos a procesos se hace de forma independiente en cada host, perro es práctica asignar procesos usados frecuentemente a sockets fijos, los cuales se les hacen conocer a los desarrolladores de aplicaciones. Los puertos menores de 1024 son denominados *puertos bien conocidos* [5] y se reservan para los servicios más frecuentemente utilizados.

1.1.2.5 Manejo de conexiones

Los mecanismos de confiabilidad y control de flujo exigen que TCP inicialice y mantenga cierta información de estado para cada flujo de datos. La combinación de esta información, incluyendo sockets, números de secuencia y tamaños de las ventanas recibe el nombre de conexión.

TCP establece una conexión al inicio de la transferencia de datos entre dos procesos y la cierra cuando la comunicación ha finalizado. El protocolo utilizado por TCP para el establecimiento de las conexiones es conocido como protocolo de acuerdo de tres vías (three-way handshake protocol) [6]. Este procedimiento se inicializa por un lado de la conexión y es respondido por el otro lado, aunque también funciona si ambos extremos intentan inicializar la conexión [4].

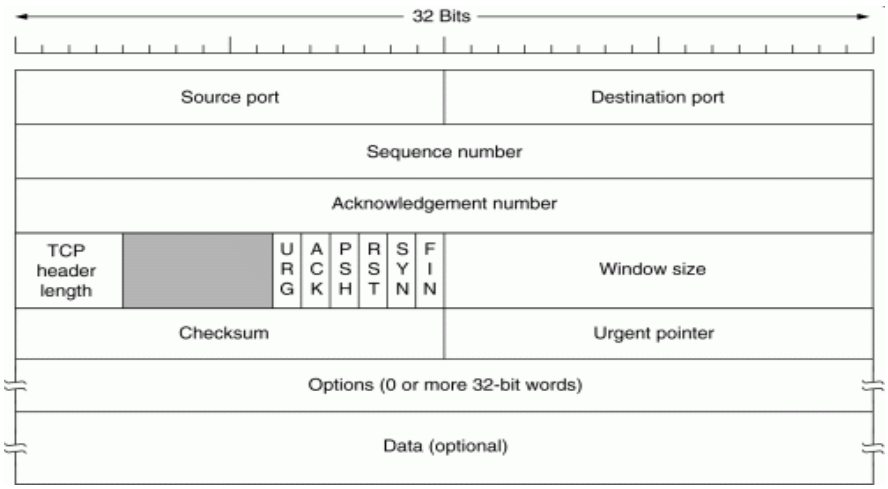


Figura No. 7. Encabezamiento del protocolo TCP [1]

1.2 El protocolo HTTP

El protocolo HTTP (Hypertext Transfer Protocol) es un protocolo de la capa de aplicación para sistemas de hipermedia colaborativos y distribuidos [7]. Este protocolo ha sido usado en el servicio de información World Wide Web desde 1990. En su primera versión (HTTP/0.9) fue un protocolo simple para la transferencia de datos en bruto a través de Internet. La versión 1.0 del protocolo [8] añadió la capacidad de que los mensajes intecambiados siguieran el formato MIME (Multipurpose Internet Mail Extensions) [9]

HTTP es un protocolo se solicitud – respuesta. Un cliente envía una solicitud al servidor indicando el método, URI (Identificador Universal de Recurso) y la versión del protocolo, seguido por el mensaje en formato MIME indicando información adicional, como algunos modificadores de la solicitud (encabezados), información acerca del cliente, etc.

El servidor responde con una línea de estado, incluyendo el mensaje de la versión del protocolo, y un código indicando error o éxito en la operación, seguido de un mensaje en formato

MIME que contiene información del servidor, seudo-información acerca del recurso y posible contenido del mismo.

En el caso más sencillo de operación del protocolo la comunicación es iniciada por el cliente, consistente en una solicitud a ser aplicada a determinado recurso en un servidor dado. La figura No. 8 ilustra tal escenario.

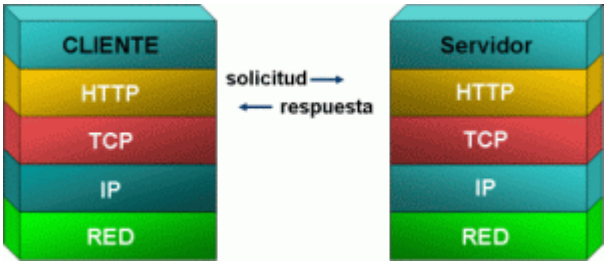


Figura No. 8 Interacción directa cliente – servidor en una solicitud HTTP

Un escenario más complejo se presenta cuando existen intermediarios entre el cliente y el servidor en la cadena solicitud – respuesta. La especificación del protocolo [7] identifica tres tipos de intermediarios: proxy, gateway y túnel. Un proxy es un agente de reenvío, recibe solicitudes de los clientes destinadas a un URI y reescribe el mensaje en su totalidad o parte de éste antes de entregarlo al servidor. Un gateway por su parte, actúa como una capa a través de la cual se acceden a otros servicios, haciendo una traducción, en caso de ser necesario, de las solicitudes HTTP al protocolo nativo del servicio; permite acceder por HTTP a otros servicios que directamente no lo soportan. El túnel actúa como relevo entre dos conexiones sin alterar el mensaje original; este intermediario se utiliza cuando la conexión pasa a través de cortafuegos, aún cuando éste no comprenda la semántica ni el contenido de los mensajes. La figura No. 9 ilustra el escenario de la comunicación entre cliente y servidor en la que existe un elemento intermedio, en este caso un proxy.

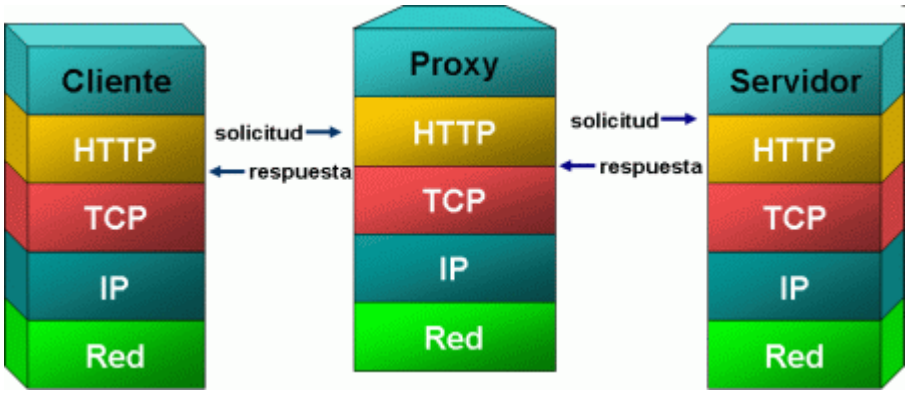


Figura No. 9 Solicitud HTTP con elemento intermedio

En términos generales, la cantidad de conexiones generadas entre el cliente y el servidor será igual a:

$$C = N + 1 \quad (2)$$

Donde: C - Cantidad de conexiones

N - Cantidad de elementos intermedios

Las opciones de la comunicación HTTP pueden aplicarse entre los elementos vecinos más cercanos, exceptuando el túnel, a los puntos extremos de la cadena solicitud – respuesta o a todas las conexiones generadas a lo largo de la cadena.

Cualquier elemento, exceptuando el túnel, puede emplear una memoria interna (cache) para manejar las solicitudes. El efecto de usar esta técnica es una disminución del tiempo de respuesta, si los participante en la comunicación a lo largo de la cadena solicitud – respuesta han almacenado en cache una repuesta aplicable a la solicitud. No todas las respuestas son susceptibles de ser almacenadas internamente, también las solicitudes pueden incluir modificadores que indiquen el comportamiento a seguir en el almacenamiento de las respuestas que ellas generan [7].

La comunicación HTTP ocurre usualmente sobre conexiones TCP/IP, el puesto utilizado por omisión es el 80 [5], pero pueden ser utilizados otros puertos; esto no impide que el protocolo pueda ser implementado sobre otros protocolos e incluso en otro tipo de redes. HTTP confía que el nivel de transporte ofrezca un servicio confiable, cualquier protocolo que suministre tal servicio puede ser utilizado.

El la versión 1.0 del protocolo, se utiliza una conexión diferente para cada solicitud generada por un cliente. La versión 1.1, aunque mantiene esta característica, adiciona la capacidad de utilizar una misma conexión para una o más solicitudes. Debido a esta particularidad se dice que el protocolo HTTP no mantiene estado; ya que cada solicitud inicia una nueva conexión, aunque las solicitudes provengan de un mismo documento hipermedia. Quiere decir esto que si en un documento existen diferentes piezas de información, por ejemplo, un texto y tres imágenes, se generarán al menos 4 conexiones, una por cada pieza de información, antes que el documento haya sido transferido por completo. Este hecho ha tenido un impacto notable en la adopción de esta plataforma para el desarrollo de aplicaciones y en las tecnologías asociadas, lo cual ha dado como resultado la creación de varias técnicas para solucionar el mismo.

1.2.1 IDENTIFICACIÓN DE RECURSOS

Para el protocolo HTTP, un identificador universal de recurso (URI) es una cadena de caracteres con formato que identifica a través de un nombre, ubicación o cualquier otra característica un recurso. HTTP utiliza un caso particular de URI denominado URL (Localizador Universal de Recursos) [10]. Estos se utilizan para hallar un recurso basado en una identificación abstracta de la ubicación del mismo. Una vez ubicado el recurso a través de su URL, el sistema puede realizar diversas operaciones sobre éste.

1.2.1.1 LOCALIZADOR UNIVERSAL DE RECURSOS (URL)

La RFC 1738 define la sintaxis del URL como:

<scheme>:<scheme-specific-part>

Donde:

- ✓ scheme: esquema a ser usado (ver Tabla No. 4
- ✓ scheme-specific-part: contiene información propia del esquema utilizado.

El nombre del esquema puede incluir caracteres mayúsculos y minúsculos de “a” a “z”, dígitos y los caracteres “+”, “.” y “-“. Los programas que utilizan URL, por flexibilidad, no deben hacer distinción entre mayúsculas y minúsculas.

Muchos esquemas de URL confieren un significado especial a determinados caracteres, la aparición de éstos en la parte específica del esquema tiene una semántica específica; éstos caracteres incluyen ";", "/", "?", ":", "@", "=".

Tabla No. 4. Esquemas más utilizados y sus protocolos asociados

Esquema	Protocolo
ftp	FTP
http	HTTP
mailto	SMTP
news	NNTP
telnet	TELNET
file	Acceso a archivos locales

Mientras el resto del URL puede variar en dependencia de un esquema en particular muchos de los esquemas usan un formato similar:

//<user>:<password>@<host>:<port>/<url-path>

Donde:

- ✓ user: Nombre del usuario que accede al recurso
- ✓ password: Contraseña
- ✓ host: Nombre del host totalmente calificado FQDN [11] o su dirección IP
- ✓ puerto: Puerto del socket en la computadora donde se ejecuta el servicio [5]
- ✓ url-path: Información propia del esquema

1.2.1.2 El esquema http

La sintaxis del URL para el esquema http se define como:

http_URL = "http:" "//" host [":" port] [abs_path ["?" query]]

Si el Puerto no viene indicado se asume el Puerto 80. El URL del esquema HTTP identifica a un recurso que se encuentra ubicado en el servidor HTTP que está escuchando conexiones por el puerto especificado en el host indicado.

1.2.2 Operaciones del protocolo HTTP

El protocolo HTTP define un conjunto de operaciones [7] a realizar sobre un recurso identificado por un URL, aunque no todas las implementaciones las incluyen todas.

OPTIONS.

Representa una solicitud de información acerca de las opciones de comunicación disponibles en la cadena solicitud respuesta para el recurso. Permite determinar las opciones y los requerimientos asociados al mismo o las capacidades del servidor, sin que implique la transferencia del recurso.

Las respuestas a esta operación no son almacenables.

Si el recurso solicitado esta indicado como “*” las respuesta a la solicitud se aplicará al servidor en general y no a un recuso en particular. Debido a que las opciones de comunicación del servidor dependen del recurso al que se desee acceder, esta alternativa funciona solamente como un método del tipo “ping”, o sea, para comprobar el grado de reacción del servidor.

GET

Recupera la información contenida en el recurso identificado en el URL, la información es transferida al cliente. Este método tiene dos alternativas:

GET condicional: Recupera la información si el recurso satisface los requerimientos indicados en los campos de encabezamiento *If-Modified-Since*, *If-Unmodified-Since*, *If-Match*, *If-None-Match* o *If-Range*, presentes en el mensaje de solicitud. Este método puede reducir el tráfico en la red permitiendo que los proxies actualicen sus estructuras de almacenamiento temporales eficientemente y evitando transmitir información al cliente que no ha cambiado desde su última recuperación.

GET parcial: Este método no recupera la información del recurso en su totalidad, sino que utiliza el campo de encabezamiento *Range* para conocer que porción del recurso recuperar en la solicitud. Al igual que el GET condicional, permite reducir el tráfico en la red evitando transferir al cliente información que ya éste posee y que no ha sido alterada en el servidor.

Las respuestas a la solicitud GET pueden ser almacenables

HEAD

El método HEAD produce una respuesta idéntica a la del método GET, excepto que la información no es transferida físicamente del servidor al cliente. Se utiliza para obtener información del recurso sin solicitar la transferencia en sí, es útil en el proceso de validación de los enlaces en los hipertextos, chequeos de accesibilidad y modificación reciente.

Un Proxy puede utilizar este método para actualizar por si mismo la versión almacenada de los recursos que tiene almacenados sin que el cliente haya solicitado una transferencia.

POST

Este método se utiliza para indicar al servidor que acepte el recurso que se envía incluido en la solicitud como un recurso subordinado al que se identifica en el URL. Fue diseñado para:

- ✓ Interactuar con servicios de Internet como los grupos de noticias, correo electrónico, etc.
- ✓ Enviar bloques de datos, como los contenidos en un formulario, a un proceso de manipulación de datos.
- ✓ Interactuar con bases de datos.

PUT

Indica que el recurso que se encuentra incluido en la solicitud sea almacenado físicamente en el servidor en el lugar indicado por el URL. De existir un recurso en esa ubicación, el nuevo recurso se interpretará como una nueva versión modificada del existente, en caso contrario, si el servidor tiene acceso a crear un recurso en la ubicación especifi-

cada intentará hacerlo e informará al cliente mediante un código de estado sobre el resultado de esta operación.

Aunque este método y el anterior envían información al servidor, difieren fundamentalmente en la interpretación de los datos que se envían incluidos en el mensaje de la solicitud. En POST los datos enviados deben ser procesados por el recurso identificado en el URL, mientras que en PUT los datos constituyen en sí el recurso que se intentará crear.

DELETE

Como su nombre lo indica, este método solicita al servidor que elimine el recurso indicado en el URL. Esta operación puede eliminar físicamente el recurso o moverlo a una ubicación inaccesible por el cliente haciéndolo invisible al servicio.

TRACE

Permite que el cliente conozca lo que está sucediendo en el otro extremo de la cadena solicitud respuesta y utilice esa información para chequeos y diagnósticos.

CONNECT

La especificación 1.1 del protocolo HTTP reserva este método para ser utilizado en proxies que pueden dinámicamente conmutar a modo túnel

1.2.3 ENCABEZADOS DE LA SOLICITUD

Los encabezados de la solicitud forman parte del mensaje de solicitud enviado por el cliente al servidor y aparecen después de la línea que indica el método, el URI y la versión del protocolo. Indican las capacidades del cliente y también información que el servidor debe tener en cuenta en el momento de transferir el recurso solicitado. A continuación se relacionan los encabezados más utilizados y soportados por los clientes (navegadores). El valor de muchos de estos encabezados depende del navegador en cuestión y de cómo haya sido configurado éste. [12].

ACCEPT: Indica los tipos de datos que el cliente puede manejar. El anexo 1 muestra los tipos (en formato MIME) más comunes

ACCEPT-CHARSET: Indica el juego de caracteres que el cliente acepta.

ACCEPT-ENCODING: Indica el tipo de codificación que el cliente puede manejar

ACCEPT-LANGUAJE: Indica los idiomas que el cliente acepta

AUTHORIZATION: Utilizado en la identificación de los clientes cuando acceden a recursos protegidos.

CACHE-CONTROL: Utilizado para especificar las opciones de las páginas a ser almacenadas en proxies.

CONNECTION: Indica cuando el cliente soporta conexiones persistentes.

CONTENT-LENGTH: Este encabezado solamente es aplicable alas solicitudes que utilizan el método POST e indica la cantidad de bytes del dato transferido con la solicitud.

CONTENT-TYPE: Aunque este encabezado usualmente aparece en los encabezados de la respuesta, puede aparecer también en la solicitud para indicar el tipo MIME del dato transferido utilizando métodos PUT y POST.

COOKIES: Utilizado para enviar *cookies* al servidor que previamente han sido transferidos al cliente.

EXPECT: Indica al servidor que el cliente desea enviar un documento anexo y desea saber si el servidor lo aceptará.

FROM: Indica la dirección de correo electrónico del responsable de la solicitud.

HOST: Indica el host y el puerto, tal como viene indicado en el URL. Útil en el reenvío de solicitudes y en host que tienen múltiples nombres.

IF-MODIFIED-SINCE: Indica que el cliente solo está interesado en un recurso que haya sido modificado después de la fecha indicada en este encabezado.

IF-UNMODIFIED-SINCE: Es lo opuesto al anterior, indica que el cliente esta interesado en el recurso solo si la fecha de modificación de éste es anterior a la fecha indicada.

PRAGMA: Si la comunicación se está realizando a través del proxy indica que el cliente está interesado en el recurso localizado en el sitio original, aún cuando el proxy tenga una versión actualizada almacenada del mismo.

PROXY-AUTHORIZATION: Utilizado por los clientes para identificarse ante los proxies que lo requieran.

RANGE: Indica que el cliente tiene una copia parcial del recurso y está solicitante la información restante

REFERER: Este encabezado indica el recurso que ha provocado que éste sea solicitado. Por ejemplo, si en el documento hipermedia 1 existe un enlace que conduce al documento hipermedia 2, el URL completo del primer documento se incluye en la solicitud del documento 2 a través de este encabezado.

USER-AGENT: Identifica al cliente, nombre del producto, versión, sistema operativo, etc.

1.2.4. ENCABEZADOS DE LA RESPUESTA

Los encabezados de la respuesta, conjuntamente con los códigos de estado, se incluyen en la respuesta que envía el servidor al cliente luego de haber procesado una solicitud. Los encabezados más utilizados y su significado se relacionan a continuación [12].

ACCEPT-RANGE: Informa al cliente si el servidor acepta solicitudes con el encabezado RANGE

AGE: Utilizado por los proxies para informar el tiempo que lleva de generado el documento en el servidor original.

ALLOW: Informa al cliente los métodos que el servidor soporta (GET, PUT, POST, etc.).

CACHE-CONTROL: Indica al cliente las circunstancias en las que el documento puede ser almecenado. Un valor "*public*" indica que el documento puede ser almacenado, "*private*" indica que el documento puede ser almacenado en estructuras de almacenamiento privadas, no compartidas; "no-cache" indica que el documento no debe ser almacenado en proxies, "no-store" indica que el documento no debe ser almacenado en proxies ni localmente en el cliente, "max-age=xxx" indica que el documento es considerado obsoleto después de xxx segundos, "s-max-age=xxx" indica que las estructura de almacenamiento compartido deben considerar este documento como obsoleto después de xxx segundos.

CONNECTION: Informa al cliente la capacidad de utilizar conexiones persistentes.

CONTENT-ENCODING: Indica la forma en que la página fue codificada durante la transmisión.

CONTENT-LANGUAGE: indica el idioma en que fue escrito el documento según la RFC 1676.

CONTENT-LENGTH: Indica la cantidad en bytes de la respuesta.

CONTENT-MD5: Indica una secuencia MD5 para el documento.

CONTENT-RANGE: Indica la cantidad de bytes que se han enviado en una solicitud parcial.

CONTENT-TYPE: Indica el tipo MIME del recurso enviado (ver Anexo I).

DATE: Indica la fecha en que se generó la respuesta.

EXPIRES: Indica el tiempo en segundos transcurridos los cuales el documento debe ser considerado obsoleto.

LAST-MODIFIED: Fecha de la última modificación del recurso.

LOCATION: Notifica al cliente un URL. Al recibir esta respuesta el cliente automáticamente solicita el recurso indicado por el nuevo URL. Se utiliza para redireccionar al cliente hacia otro URL.

REFRESH: Cantidad de segundos que debe esperar el cliente antes de refrescar automáticamente el recurso utilizando URL original.

SET-COOKIE: Especifica un *cookie* asociado al recurso.

WWW-AUTHENTICATE: Indica al cliente que para acceder al recurso a solicitud debe incluir el encabezado AUTHORIZATION. Al recibir esta respuesta el cliente debe solicitar al usuario la información necesaria a ser incluida en su encabezado.

1.2.5 CÓDIGOS DE ESTADO

El código de estado forma parte de la respuesta enviada por el servidor al cliente y se utiliza para indicar el éxito o fracaso de la operación solicitada expresados en valores numéricos, los cuales se encuentran agrupados en 5 categorías:

100 – 199: Son códigos de información, indican al cliente que sebe responder con otra acción

200 – 299: Indican que la solicitud ha sido procesada con éxito.

300 – 399: Indica que el recurso ya no está disponible en la ubicación indicada en el URL. Usualmente van acompañados de un encabezado LOCATION.

400 – 499: Indican error por el cliente.

500 – 599 Indican error por el servidor.

El anexo II recoge los códigos de estado más utilizados y su significado.

1.3 LOS SISTEMAS DE CONTROL DE SUPERVISIÓN Y ADQUISICIÓN DE DATOS

Los sistemas de control de supervisión y adquisición de datos, más conocidos por sus siglas en inglés SCADA (supervisory control and data acquisition) son sistemas computacionales que permiten obtener y analizar datos en tiempo real asociados a un proceso determinado. Estos sistemas obtienen información de los dispositivos que controlan los procesos y la transfieren a un servidor central para su procesamiento y almacenamiento. También alertan sobre de condiciones anormales que detectan en el proceso que es objeto del monitoreo. Su modo de operación puede ser de dos tipos: Maestro/esclavo y/o RBE (Report by Exception).

Como su nombre lo indica, no constituyen un sistema de control completo, sino que centran su atención en la etapa de supervisión, por lo que están constituidos completamente por software y pueden encontrase disponibles para una variedad de plataformas y sistemas operativos tales como Windows y cualquier variante de los sistemas UNIX.

1.3.1 ARQUITECTURA DE LOS SISTEMAS SCADA

En los sistemas SCADA se aprecian dos capas bien definidas: la capa “cliente”, la cual se encarga de la interacción hombre-máquina (HMI, de sus siglas en inglés) y la capa “servidor de datos”, sobre la cual recae el mayor peso del tratamiento de la información en este tipo de sistema. Los servidores de datos se comunican los dispositivos que controlan los procesos (los cuales pueden ser PLC, RTU, etc.) a través de las redes de comunicación de datos o mediante buses de campo los cuales pueden ser propietarios o no. La figura No. 10 muestra la arquitectura típica de estos sistemas.

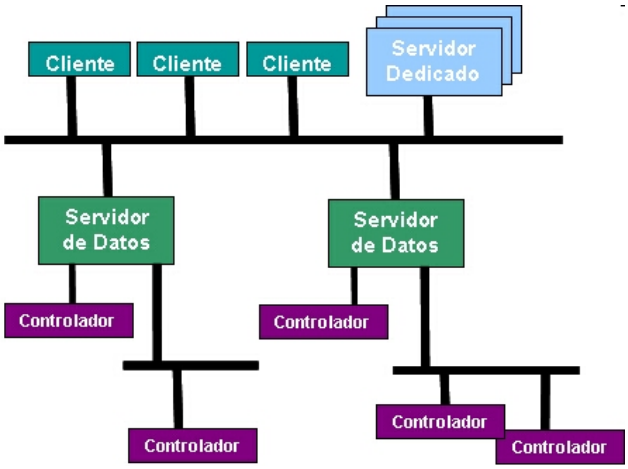


Figura No. 10. Arquitectura de los sistemas SCADA

Esta tipo de sistemas están caracterizados por utilizar técnicas de programación concurrente debido a que deben realizar varias tareas al mismo tiempo, así como por la utilización de servidores de bases de datos para el almacenamiento de la información obtenida. La figura No. 11 muestra la arquitectura de software de sistemas SCADA mas citada en las publicaciones científicas.

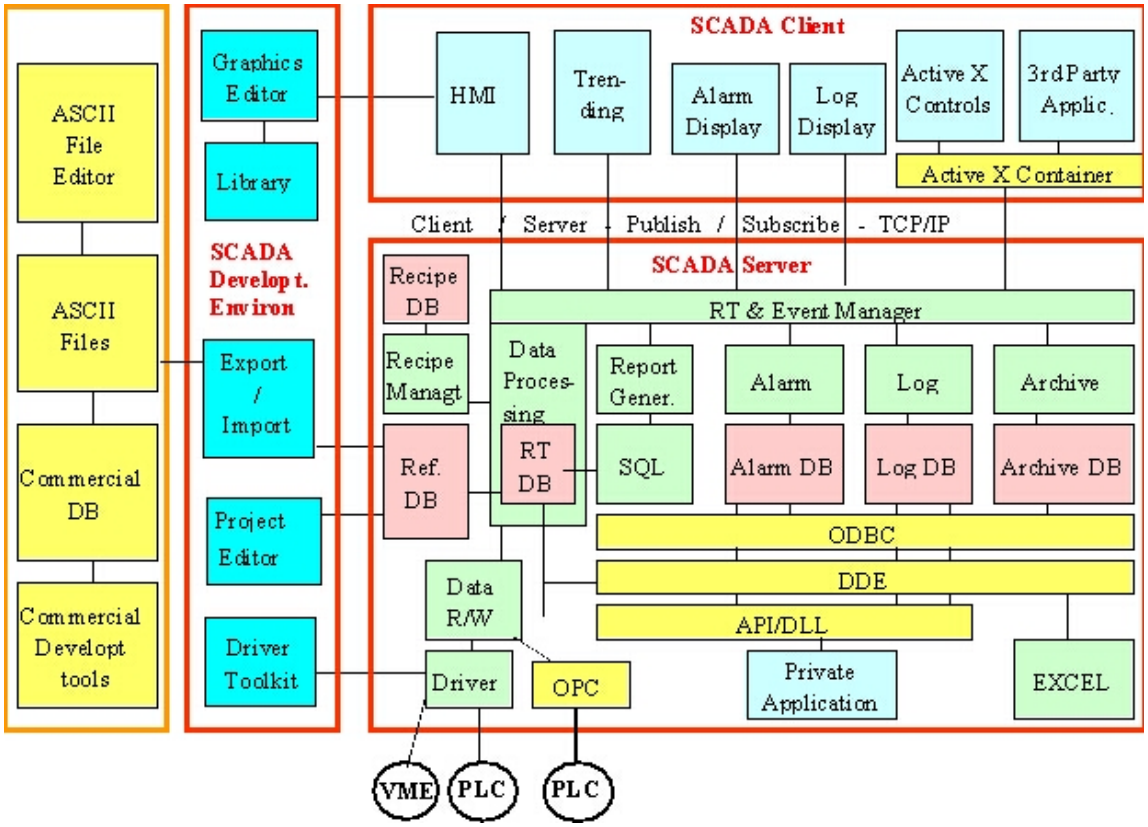


Figura No 11. Arquitectura de software de los sistemas SCADA [13]

1.3.2 CARACTERÍSTICAS FUNCIONALES

Los servidores de datos (según la figura No. 10) son los encargados de encuestar a los dispositivos que controlan los procesos, chequear las alarmas, almacenar los datos y dejar trazas de la ejecución. Estas tareas son llevadas a cabo diferentes módulos o subsistemas, los cuales se comunican entre si utilizando formas estandarizadas o propietarias.

1.3.2.1 COMUNICACIÓN

La comunicación en estos sistemas puede dividirse en dos tipos: la comunicación entre clientes y servidores y entre servidores y dispositivos de control. En el primer caso se utiliza generalmente protocolos de comunicación disponibles para la tecnología de red de área local que se utilice. En la actualidad esta comunicación se realiza mayoritariamente a través del protocolo TCP/IP.

La comunicación entre servidores de datos y dispositivos de control se ha realizado tradicionalmente a través de protocolos de comunicación de bajo nivel, propietarios en algunos casos y en otros estándares establecidos, pero en la actualidad se evidencia una tendencia a la

adopción de TCP/IP como el protocolo de comunicación por excelencia en ambos entornos. El anexo III muestra un dispositivo de control que incorpora TCP/IP y HTTP como protocolos de comunicación con el servidor de datos.

Los servidores de datos encuestan a los dispositivos de control (modo de operación Maestro/esclavo) a intervalos de tiempo para obtener información del estado de los parámetros de los procesos que éstos controlan. Estos intervalos pueden ser uniformes o diferentes para cada parámetro. Para notificar las condiciones anormales (alarmas) generalmente se utiliza el modo de operación RBE; de esta forma el dispositivo es el encargado de notificar al servidor la ocurrencia de una condición extrema.

Estos tipos de sistemas por lo general suministran controladores de software (drivers) que soportan los protocolos de comunicación de una gran variedad de controladores de procesos, que como se mencionó anteriormente, pueden ser propietarios o no. Opcionalmente suministran también herramientas que permiten crear controladores de software para protocolos de comunicación que no hayan sido incluidos como el producto, a tales herramientas se les conoce como "*Drivers Development Kit*".

1.3.2.2 INTERACCIÓN CON OTROS PRODUCTOS

La interacción con otros productos hasta la fecha se había hecho generalmente utilizando técnicas propietarias, en la actualidad el sector de la automatización se pronuncia por la adopción de estándares en lo que se refiere a la conectividad de los diferentes módulos que pueden conformar un sistema de este tipo.

En este sentido han emergido alternativas para garantizar esa conectividad, tal es el caso de OPC (OLE for Process Control). OPC propone lograr una conectividad abierta a través de estándares abiertos [14]. La interoperabilidad se logra mediante la creación y mantenimiento de especificaciones, de las cuales se cuenta en la actualidad con siete que están en estado final o en desarrollo; que satisfacen necesidades específicas de la industria; las mismas tienen su basamento en estándares y tecnologías utilizados en el mercado computacional.

El primer estándar surgido, llamado en la actualidad "*Especificación de acceso a datos*" fue el producto de la cooperación de un grupo de empresas líderes a nivel mundial en el mercado de las soluciones de automatización y la empresa Microsoft. La especificación (originalmente ba-

sada en la tecnología COM y DCOM) incluía un conjunto estándar de objetos, interfaces y métodos a usar en el control de procesos y en las aplicaciones.

La analogía más comúnmente utilizada para explicar el rol de OPC dentro del mundo de las soluciones de automatización es la del soporte a los servicios de impresión en sistemas operativos como MS DOS y Windows. En MS DOS, cada productor de software debía preocuparse por suministrar los controladores de impresión para cada tipo de impresora a la cual deseaba dar soporte en su producto. Con este enfoque, para un tipo de impresora dado podían existir tantos controladores como aplicaciones dieran soporte para ella. En el sistema operativo Windows el soporte de impresión es un servicio del sistema operativo y la responsabilidad de la construcción de los controladores de impresión para un tipo de impresora dado no recae en los productores de software sino en el productor de la impresora en si.

En el mundo de las soluciones de automatización, cada productor elaboraba el software y los controladores para acceder a los dispositivos de control que incorporara (PLC, etc.). La tecnología OLE de Windows permitió añadir estandarización, en la actualidad los productores de dispositivos de control elaboran servidores de acceso a datos conformes con OPC y los clientes (como los SCADA) se convierten en clientes OPC.

A pesar de que la primera especificación solo contemplaba el acceso a datos de campo, rápidamente surgió la necesidad de estandarizar el acceso a otros tipos de datos como las alarmas, datos históricos y procesamiento en lotes. En la actualidad, entre las especificaciones actuales y emergentes se encuentran [14]:

- ✓ OPC Data Access: Esta especificación, que contemplaba el transporte de datos desde los dispositivos de control hasta las aplicaciones en la actualidad se encuentra en un proceso de evolución. Se trabaja en la especificación Data Access 3, la cual incorpora esquemas XML para definir los datos.
- ✓ OPC Alarms & Events: Se centra en las alarmas y los procesos de notificación a demanda. Incluye alarmas de procesos, acciones de operador, mensajes informacionales, mensajes de auditorias y trazas, entre otros.
- ✓ OPC Batch: Se encarga de definir estándares para los procesos por lotes. Incluye interfaces para el intercambio de información acerca de las capacidades de los equipos y las condiciones de operación.

- ✓ OPC Data Exchange: Se centra en la comunicación entre servidores a través de redes fieldbus Ethernet. Añade configuración remota, diagnóstico así como servicios de monitoreo y gestión.
- ✓ OPC Historical Data Access: Se encarga de estandarizar el acceso a datos ya almacenados.
- ✓ OPC Security: Especifica como controlar el acceso de los clientes a los servidores de datos para proteger la información sensible y prevenir modificaciones no autorizadas de los parámetros de los procesos.
- ✓ OPC XML DA: Provee reglas consistentes para representar los datos de procesos en el lenguaje XML, permitiendo la integración con tecnologías como los servicios Web.

Para certificar la compatibilidad de un producto dado con OPC se han creado pruebas de compatibilidad, las cuales permiten la certificación de un determinado producto como compatible OPC.

CAPÍTULO II. TECNOLOGÍAS DE DESARROLLO DE APLICACIONES WEB

No queda duda en los momentos actuales que Internet ha revolucionado la forma de hacer las cosas, comenzando desde la comunicación, la búsqueda de información, el trabajo colaborativo, hasta el desarrollo y ejecución de aplicaciones. En ese proceso algunas herramientas y servicios han tenido un marcado protagonismo y una de ellas es sin dudas el World Wide Web (WWW).

La Web nació alrededor de 1989 a partir de un proyecto del CERN, en el que Tim Berners-Lee construyó el prototipo que dio lugar al núcleo de lo que hoy es la World Wide Web. La intención original era hacer más fácil el compartir textos de investigación entre científicos y permitir al lector revisar las referencias de un artículo mientras lo fuera leyendo. Un sistema de hipertexto enlazaría todos los documentos entre sí para que el lector pudiera revisar las referencias de un artículo mientras lo fuera leyendo. El nombre original del prototipo era "Enquire Within Upon Everything". El programa inicial del CERN, "WWW", sólo presentaba texto, pero navegadores posteriores añadieron la capacidad de presentar también gráficos. El imparable avance técnico de la WWW permite hoy incluso servicios en tiempo real como webcasts, radio web y webcams en directo.

Finalmente, debido a la evolución constante de los navegadores, la necesidad cada vez más de mostrar contenido dinámico y a la ubicuidad inherente en el servicio se han trasladado el desarrollo y entorno de ejecución de aplicaciones hacia la plataforma Web.

Existen varias tecnologías que permiten el desarrollo y ejecución de aplicaciones en el entorno Web, algunas de ellas son propietarias y otras de código abierto; entre las predominantes de encuentran:

- ✓ CGI
- ✓ ASP
- ✓ PHP
- ✓ ASP.NET
- ✓ JAVA

2.1 CGI

Common Gateway Interface (Pasarela de Interfaz Común) es una importante tecnología que permite a un cliente (navegador Web) solicitar datos de un programa ejecutado en un servidor Web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor Web y una aplicación externa [18].

Las aplicaciones CGI fueron una de las primeras maneras prácticas de crear contenido dinámico para las páginas Web. En una aplicación CGI, el servidor Web pasa las solicitudes del cliente a un programa externo, la figura No. 12 muestra esta interacción. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional. CGI ha hecho posible la implementación de funciones nuevas y variadas en las páginas Web, de tal manera que esta interfaz rápidamente se volvió un estándar, siendo implementada en todo tipo de servidores Web.

Figura No. 12 Comunicación entre el navegador, servidor Web y la aplicación CGI

El CGI define dos mecanismos para pasar datos desde el servicio WWW a una aplicación externa, el primero utiliza variables de entorno y el segundo la entrada estándar. La recuperación de resultados se produce siempre a través de la salida estándar. El primer método recibe el nombre de método GET y el segundo método se denomina POST. Junto con las opciones del servidor el protocolo CGI se basa también en los mecanismos de los clientes para introducir información.

2.1.1 FORMULARIOS

Los formularios son un elemento esencial de las aplicaciones Web, se pueden encontrar en cualquier ámbito, desde el proceso de registro para poder acceder a algún servicio o propiamente dentro de una aplicación que gestione datos en este entorno. Los formularios permiten introducir en un cliente WWW información estructurada que puede utilizarse como datos de entrada de una aplicación CGI. Para el soporte de formularios el lenguaje HTML permite la definición de de distintos tipos de campos.

Un formulario en HTML se limita con los marcadores <*FORM*> y <*/FORM*>. Un formulario puede incluirse en cualquier parte de un documento HTML, y sobre un mismo documento pueden insertarse distintos formularios.

Los formularios tienen un atributo denominado *METHOD* mediante el cual se indica de que forma se enviarán los datos al servidor, sus valores pueden ser *POST* y *GET.*

2.1.2 EL MÉTODO GET

En el método GET la aplicación CGI recibe la información a través de variables de entorno. El proceso de lanzamiento de una aplicación CGI con el método GET atraviesa por los siguientes pasos.

1. El cliente WWW solicita un servicio de una aplicación CGI.
2. El servidor HTTPD recibe la solicitud y los datos de entrada.
3. El servidor crea un entorno y crea variables en el con los datos de entrada.
4. El servidor ejecuta la aplicación CGI en este entorno.
5. La aplicación CGI procesa las variables de entorno y recupera los datos de entrada.
6. La aplicación CGI se ejecuta produciendo un resultado sobre su salida estándar.
7. EL servidor HTTP redirecciona la salida estándar de la aplicación CGI hacia el cliente WWW.
8. El cliente WWW recibe el resultado de su consulta.

2.1.3 EL MÉTODO POST

El método POST es el método recomendado para el paso de información de formulario a una aplicación CGI. En este método la información se pasa a través de la entrada estándar de la aplicación CGI. El proceso de lanzamiento de una aplicación CGI con el método POST es el siguiente:

1. El cliente WWW solicita un servicio de una aplicación CGI.
2. El servidor HTTPD recibe la solicitud y los datos de entrada.
3. El servidor ejecuta la aplicación CGI pasándole la información a través de la entrada estándar.
4. La aplicación CGI procesa su entrada estándar y recupera los datos de entrada.
5. La aplicación CGI se ejecuta produciendo un resultado sobre su salida estándar.

6. EL servidor HTTP redirecciona la salida estándar de la aplicación CGI hacia el cliente WWW.
7. El cliente WWW recibe el resultado de su consulta.

A pesar que CGI permitió (y aún lo hace) incorporar contenido dinámico esta técnica tiene serios inconvenientes relacionados con el rendimiento del servidor Web, que se ve forzado a lanzar una aplicación CGI por cada solicitud que recibe. Un servidor que reciba una gran cantidad de solicitudes puede ver afectado seriamente su rendimiento por el uso de esa técnica.

Para solucionar ese problema surgió FastCGI, el cual es una alternativa al CGI estándar, cuya diferencia radica principalmente en el hecho de que el servidor crea un único proceso persistente por cada programa FastCGI en lugar de por cada solicitud del cliente.
Entre los inconvenientes que presenta la técnica CGI (además del mencionado anteriormente) se encuentra que la respuesta que recibe el navegador es generada completamente por la aplicación CGI, la cual se escribe en un lenguaje de programación, que puede ser interpretado (Perl, etc.) o compilado (C, C++, etc.). Esto hace necesario que la respuesta, un documento HTML, se genere desde un lenguaje de programación y hace bastante engorroso la creación de aplicaciones complejas basadas en esta técnica.

La solución apareció cuando empezaron a aparecer técnicas que permitían incrustar el código que producía el contenido dinámico dentro de un documento HTML. Esto posibilitó que diseñadores Web y programadores trabajaran cada uno en su ámbito y de forma colaborativa produjeran documentos HTML dinámicos con alta calidad. Entre estas técnicas se encuentran ASP y PHP.

2.2 ASP

Active Server Pages (**ASP**) es una tecnología del lado servidor de Microsoft para páginas Web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS). ASP ha pasado por cuatro iteraciones mayores, ASP 1.0 (distribuido con IIS 3.0), ASP 2.0 (distribuido con IIS 4.0), ASP 3.0 (distribuido con IIS 5.0) y ASP.NET (parte de la plataforma .NET de Microsoft). Las versiones anteriores a .NET se denominan actualmente (desde 2002) como ASP *clásico*.

En el último ASP clásico, ASP 3.0, hay seis objetos integrados disponibles para el programador, *Application*, *ASPError*, *Request*, *Response*, *Server* y *Session*. Cada objeto corresponde a

un grupo de funcionalidades frecuentemente usadas y útiles para crear páginas Web dinámicas.

Una página ASP está formada por código HTML con fragmentos de código escritos en Visual Basic Scritpt o J Script insertados dentro de ella. Cuando el servidor recibe una solicitud de una página ASP utiliza un módulo para interpretar el contenido de la página y generar el documento HTML resultante, el cual será enviado al navegador y será finalmente lo que recibirá el usuario.

La figura No. 13 muestra la arquitectura de la tecnología ASP.

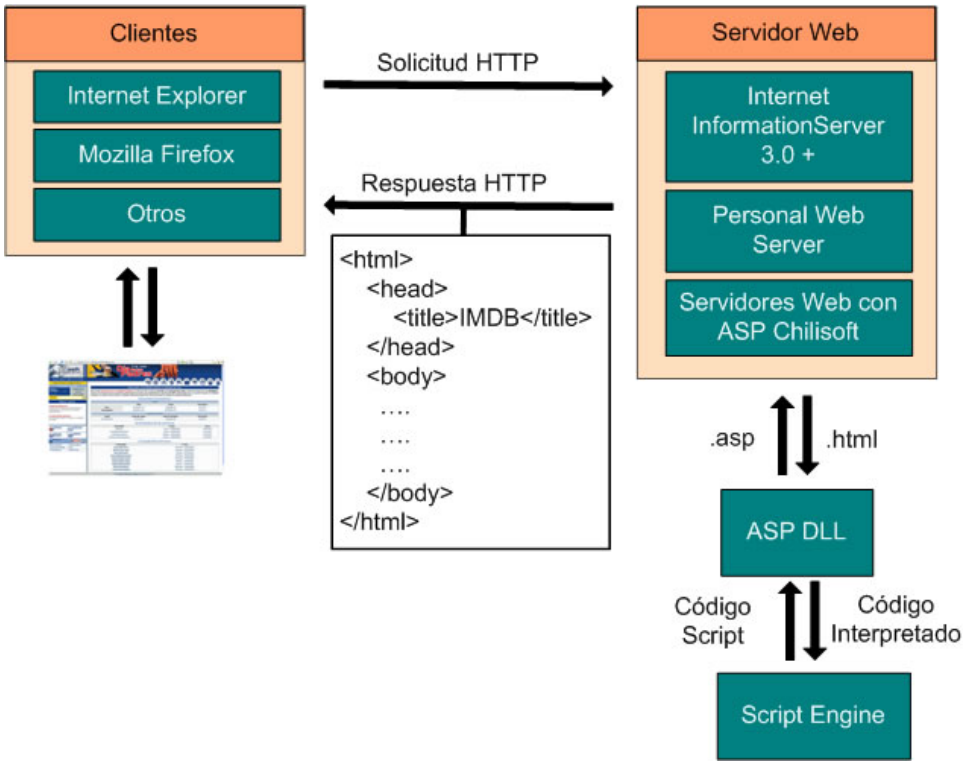


Figura No. 13 Arquitectura de la tecnología ASP

2.3 PHP

PHP es un lenguaje de programación usado generalmente para la creación de contenido para sitios Web. PHP (acrónimo recursivo de "**P**HP: **H**ypertext **P**reprocessor", inicialmente PHP Tools, o, *P*ersonal *H*ome *P*age Tools) es un lenguaje interpretado usado para la creación de

aplicaciones para servidores, o creación de contenido dinámico para sitios Web, y últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la librería GTK+.

El fácil uso y la similitud con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores experimentados crear aplicaciones complejas con una curva de aprendizaje muy suave. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones y prácticas.

Debido al diseño de PHP, también es posible crear aplicaciones con una interfaz gráfica para el usuario (también llamada GUI), utilizando la extensión PHP-GTK. También puede ser usado desde la Línea de comandos, al igual que otros lenguajes de programación, esta versión de PHP se llama PHP CLI (*Command Line Interface*).

Su interpretación y ejecución se da en el servidor, en el cual se encuentra almacenado el script, y el cliente sólo recibe el resultado de la ejecución. Cuando el cliente hace una petición al servidor para que le envíe una página Web, generada por un script PHP, el servidor ejecuta el intérprete de PHP, el cual procesa el script solicitado que generará el contenido de manera dinámica, pudiendo modificar el contenido a enviar, y regresa el resultado al servidor, el cual se encarga de regresárselo al cliente. Además es posible utilizar PHP para generar archivos PDF, Flash, así como imágenes en diferentes formatos, entre otras cosas.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, IBM DB2, Microsoft SQL Server, Firebird y SQLite; lo cual permite la creación de aplicaciones Web muy robustas.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX (y de ese tipo, como Linux), Windows y Mac OS X, y puede interactuar con los servidores Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

El modelo PHP puede ser visto como una alternativa al sistema de Microsoft que utiliza ASP.NET/C#/VB.NET, a ColdFusion de la compañía Macromedia (Actualmente Adobe), a JSP/Java de Sun Microsystems, y al famoso CGI/Perl. Aunque su creación y desarrollo se da

en el ámbito de los sistemas libres, bajo la licencia GNU, existe además un compilador comercial llamado Zend Optimizer.

PHP fue originalmente diseñado en Perl, seguidos por la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador Danés-Canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vitae y guardar ciertos datos, como la cantidad de tráfico que su página Web recibía. El 8 de junio del 1995 fue publicado "**P**ersonal **H**ome **P**age Tools" luego de que Lerdorf lo combinara con su propio *Form Interpreter* para crear PHP/FI.

Dos programadores israelíes, Zeev Suraski y Andi Gutmans, reescribieron el analizador gramatical en el año 1997 y crearon la base del PHP 3, cambiando el nombre del lenguaje a la forma actual. Experimentaciones públicas de PHP 3 comenzaron inmediatamente y fue lanzado oficialmente en junio del 1998.

Para 1999, Suraski y Gutmans reescribieron el código de PHP, produciendo lo que hoy se conoce como Zend Engine o motor Zend. También conformaron a Zend Technologies (http://www.zend.com) en Ratmat Gan, Israel. En mayo de 2000 PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El 13 de julio de 2004, PHP 5 fue lanzado, utilizando el motor Zend Engine II (o Zend Engine 2). La versión más reciente de PHP es la 5.1, que incluye el novedoso PDO (Objetos de Información de PHP o PHP Data Objects) y mejoras utilizando las ventajas que provee el nuevo Zend Engine 2.

Las ventajas de PHP pueden resumirse como:

- ✓ Lenguaje multiplataforma.
- ✓ Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- ✓ Leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML
- ✓ Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (extensiones).
- ✓ Posee una muy buena documentación en su página oficial (http://www.php.net/)
- ✓ Es gratuito y de código abierto, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Permite las técnicas de Programación Orientada a Objetos.

2.4 ASP.NET

.NET es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de la plataforma de ejecución y que permita un rápido desarrollo de aplicaciones. Basado en ésta, Microsoft intenta desarrollar una estrategia horizontal que integre todos sus productos, desde el Sistema Operativo hasta las herramientas de mercado [18].

.NET podría considerarse una respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Sun Microsystems.

A largo plazo Microsoft pretende reemplazar la Interfaz de Programación de Aplicaciones (API por sus siglas en inglés) Win32 o Windows API con la plataforma .NET. Esto debido a que la API Win32 o Windows API fue desarrollada sobre la marcha, careciendo de documentación detallada, uniformidad y cohesión entre sus distintos componentes, provocando múltiples problemas en el desarrollo de aplicaciones para el sistema operativo Windows. La plataforma .NET pretende solventar la mayoría de estos problemas proveyendo un conjunto único y expandible con facilidad, de bloques interconectados, diseñados de forma uniforme y bien documentados, que permitan a los desarrolladores tener a mano todo lo que necesitan para producir aplicaciones sólidas. La arquitectura de ASP.NET se muestra en la figura No.14.

Debido a las ventajas que la disponibilidad de una plataforma de este tipo puede darle a las empresas de tecnología y al público en general, muchas otras empresas e instituciones se han unido a Microsoft en el desarrollo y fortalecimiento de la plataforma .Net, ya sea por medio de la implementación de la plataforma para otros sistemas operativos aparte de Windows (Proyecto Mono de Ximian/Novell para Linux/MacOS X/BSD/Solaris), el desarrollo de lenguajes de programación adicionales para la plataforma (ANSI C de la Universidad de Princeton, NetCOBOL de Fujitsu, Delphi de Borland, entre otros) o la creación de bloques adicionales para la plataforma (como controles, componentes y bibliotecas de clases adicionales); siendo algunas de ellas iniciativas de distribución gratuita bajo la licencia GNU.

Con esta plataforma Microsoft incursiona de lleno en el campo de los Servicios Web y establece el XML como norma en el transporte de información en sus productos y lo promociona como tal en los sistemas desarrollados utilizando sus herramientas.

.NET intenta ofrecer una manera rápida y económica pero a la vez segura y robusta de desarrollar aplicaciones - o como la misma plataforma las denomina, soluciones - permitiendo a su vez una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier tipo de dispositivo.

ASP.NET provee servicios que permiten el desarrollo, despliegue y ejecución de aplicaciones Web y servicios Web. Al igual que su antecesor ASP es una tecnología que funciona en el lado del servidor, sustentada en formularios Web (Web Forms), los cuales han sido diseñados cuidadosamente de forma tal que la construcción de aplicaciones basadas en Web sea tan sencilla como la construcción de aplicaciones utilizando Visual Basic.

Entre las características distintivas de ASP.NET se encuentran que reutiliza las partes buenas de ASP y mejora las otras, soporta múltiple lenguajes compilados, rápido, escalable, gestionable, seguro y con soporte de herramientas, se sustenta en el marco de trabajo .NET, y utiliza un modelo de programación sencillo.

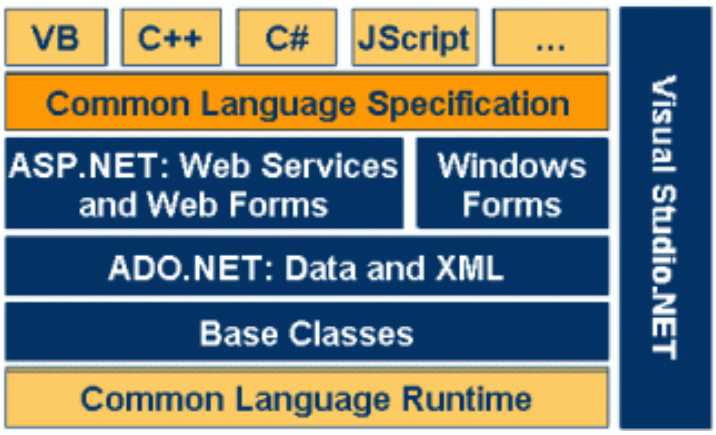


Figura No. 14. Arquitectura de ASP.NET

2.5 JAVA

Java es un lenguaje de programación orientado a objetos desarrollado por James Gosling y sus compañeros de Sun Microsystems al inicio de la década de 1990. A diferencia de los lenguajes de programación convencionales, que generalmente están diseñados para ser compilados a código nativo, Java es compilado en un bytecode que es ejecutado, por una máquina virtual.

La plataforma Java y el lenguaje Java empezaron como un proyecto interno, denominado *Sealth*, de Sun Microsystems en diciembre de 1990. El proyecto fue rebautizado como *Green*

Proyect (*Proyecto Verde*) cuando James Gosling y Mike Sheridan se unieron al equipo de trabajo. El equipo dedicó largas horas de trabajo y en el verano de 1992 tuvieron lista algunas partes de la plataforma, incluyendo el Sistema Operativo Green, el lenguaje Oak, las librerías y el hardware. La primera prueba, llevada a cabo el 3 de Septiembre de 1992, se centró en construir una PDA (Personal Digital Assistant o Asistente Digital Personal) llamada *Star7*, que contaba con una interfaz gráfica y un asistente apodado "Duke" para guiar al usuario.

En junio y julio de 1994 el equipo reorientó la plataforma hacia la Web. Sintieron que la llegada del navegador Web Mosaic, propiciaría que Internet se convirtiese en un medio interactivo, como el que pensaban era la televisión por cable. Crearon entonces un prototipo de navegador, WebRunner, que más tarde sería conocido como HotJava. En octubre de 1994, se hizo una demostración de HotJava y la plataforma Java a los ejecutivos de Sun. Java 1.0a pudo descargarse por primera vez en 1994, pero hubo que esperar al 23 de mayo de 1995, durante las conferencias de SunWorld, a que vieran la luz pública Java y HotJava. El acto estuvo acompañado por una pequeña sorpresa adicional, el anuncio de que Java sería soportado en el navegador Netscape. El 9 de enero del año siguiente, 1996, Sun fundó el grupo empresarial JavaSoft para que se encargase del desarrollo tecnológico. Dos semanas más tarde la primera versión de Java fue publicada. A partir de ese momento el crecimiento y la madurez del lenguaje y sus tecnologías y extensiones asociadas ha crecido vertiginosamente, así como la cantidad de proyectos (de código abierto o no) y empresas que han adoptado esta tecnología para su desarrollo.

Aunque desde su aparición es posible la construcción de aplicaciones para el entorno Web, mediante la creación de *Applets*, no es hasta la aparición de la edición empresarial de Java (J2EE) [20] que las empresas toman en cuanta a Java para este tipo de aplicaciones. La arquitectura de J2EE se muestra en la figura No. 15.

La plataforma Java 2 Enterprise Edition (J2EE) es fruto de la colaboración de SUN con los líderes del sector del software empresarial (IBM, Apple, Bea Systems, Oracle, Inprise, Hewlett-Packard, Novell, etc.) para definir una plataforma robusta y flexible orientada a cubrir las necesidades empresariales en e-business y business-to-business. Esta edición reduce el costo y complejidad de desarrollo deservicios multi-capas, y da por resultado servicios que pueden ser creados rápida y fácilmente.

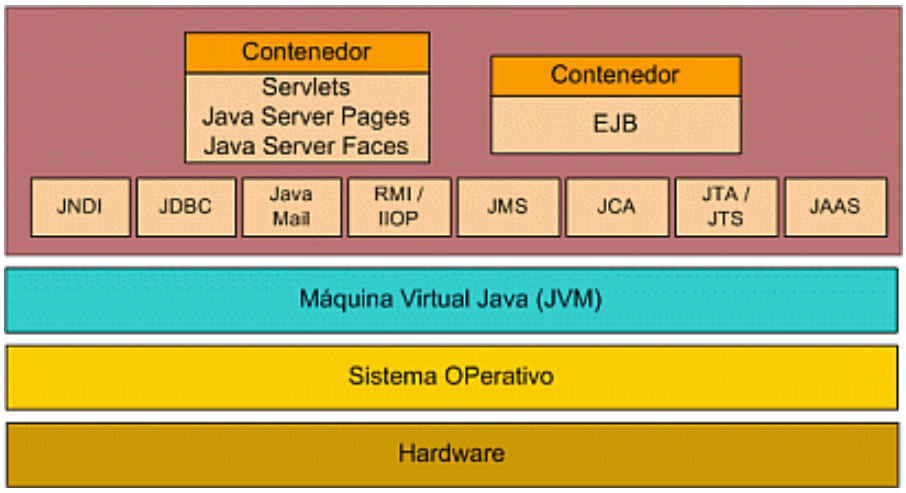


Figura No.15. Arquitectura de la edición empresarial de Java (J2EE)

Entre las tecnologías que integran la edición empresarial de Java y sobre las que se sustentan la creación de aplicaciones Web se encuentran:

- ✓ Java Server Pages
- ✓ Java Servlets

2.5.1 JAVA SERVER PAGES

La tecnología JSP, o de JavaServer Pages, es una tecnología Java que permite a los programadores generar dinámicamente HTML, XML o algún otro tipo de página Web. La misma permite al código Java y a algunas acciones predefinidas ser embebidas en el contenido estático. En las páginas JSP, se escribe el texto que va a ser devuelto en la salida (normalmente código HTML) incluyendo código java dentro de él para poder modificar o generar contenido dinámicamente. El código java se incluye dentro de las marcas de etiqueta "<%" y "%>".

La principal ventaja de JSP frente a otros lenguajes es que permite integrarse con clases Java, lo que permite separar en niveles las aplicaciones Web, almacenando en clases java las partes que consumen más recursos así como las que requieren más seguridad, y dejando la parte encargada de formatear el documento HTML en el archivo JSP. Además Java se caracteriza por ser un lenguaje que puede ejecutarse en cualquier sistema, lo que sumado a JSP le da mucha versatilidad.

Sin embargo JSP no se puede considerar un script al 100% ya que antes de ejecutarse el servidor Web compila el script y genera un Servlet, por lo tanto se puede decir que aunque este proceso sea transparente para el programador no deja de ser una aplicación compilada. La ventaja de esto es algo más de rapidez y disponer del API de Java en su totalidad.

Debido a esto la tecnología JSP, así como Java está teniendo mucho peso en el desarrollo Web profesional (sobre todo en intranets).

2.5.2 JAVA SERVLETS

Los Servlets son componentes que corren en el servidor, independientes del protocolo y de la plataforma, escritos en Java y que extienden dinámicamente la funcionalidad de los servidores. Proveen un marco de trabajo general para servicios que se basan en el paradigma solicitud – respuesta. Su primera utilización fue acceder, de forma segura, a los datos presentados utilizando HTML, y de forma interactiva consultar y modificar esos datos utilizando técnicas de generación de contenido HTML dinámico [21]. Aunque los servlets, por su definición, no están estrechamente vinculados a ningún protoloco la implementación más común esta relacionada con el protocolo HTTP, de ahí que a estos servlets se les conozca como HTTP Servlets.

Debido a que los servlets se ejecutan dentro de los servidores, no necesitan interfases gráficas de usuario, se pueden considerar como la contrapartida en el lado del servidor de los *applets*. Como parte de su funcionamiento un servlet recibe una solicitud de un cliente (en el caso mas general una solicitud HTTP), obtiene información de la solicitud, genera contenido o ejecuta procesos de la lógica de negocios teniendo en cuenta la información obtenida de la solicitud, y finalmente crea y envía una respuesta al cliente (en el caso mas general una respuesta HTTP), o redirecciona la solicitud a otro servlet.

2.6 ELECCIÓN DE LA TECNOLOGÍA A EMPLEAR EN EL DESARROLLO DEL SISTEMA DE CONTROL SUPERVISORIO

El desarrollo de aplicaciones Web multi capas requiere de funciones que deben ser ejecutadas en cada uno de los niveles que las componen:

- ✓ La capa de presentación
- ✓ La capa de lógica de negocios
- ✓ La capa de persistencia

Uno de los mayores reclamos la comunidad de desarrolladores de aplicaciones Web es que comúnmente se ven involucrados en la escritura de código que los aleja del objetivo central de la aplicación, tales como la apariencia visual de los componentes o la persistencia de los datos. Desafortunadamente en el mercado y en la comunidad de código abierto no se ven los productos que solucionen estos problemas disponibles para la mayoría de los lenguajes. Entre los lenguajes más beneficiados por el fenómeno de los marcos de trabajo (frameworks) se encuentran Java, PHP y recientemente .NET.

La capa de presentación es una de las que más esfuerzo requiere de los desarrolladores, en las tecnologías que trabajan directamente con HTML en el lado del cliente, esta capa puede concentrar hasta un 80 por ciento del tiempo de desarrollo. Java y .NET son los lenguajes mas beneficiados por la existencia de marcos de trabajo (ya sean nativos o de terceras partes) que suministran un conjunto de componentes que, de forma fácil y soportando patrones de diseño, simplifican el trabajo de la creación de interfases de usuario para el WEB.

En el caso de Java uno de los más conocidos es Java Server Faces [15]. Un marco de trabajo regido estrictamente por el patrón de diseño MVC-2, que aporta un juego de componentes lo suficientemente completo como para hacer del trabajo de creación de interfases de usuario una actividad productiva. Es de destacar también el amplio soporte que tiene este marco de trabajo en los entornos de desarrollo integrado (IDE) comerciales y de código abierto.

Existe también una gran cantidad de marcos de trabajo disponible para Java que permiten la creación de aplicaciones WEB utilizando AJAX (JavaScript Asincrónico y XML), muchas de ellas permiten el desarrollo de las aplicaciones utilizando Java Server Faces y compilan la salida que envían al navegador utilizando AJAX. Entre ellos se encuentran Apache MyFaces y OpenLaszlo.

.NET por su parte presenta los controles WEB, los cuales son parte integral de la tecnología, totalmente soportados por el entorno de desarrollo integrado de Microsoft (Visual Studio .NET), y que también aportan un juego bastante amplio de componentes para la construcción de interfases de usuario para el WEB de una forma productiva.

Para la capa de presentación se eligió el marco de trabajo Open Laszlo, el cual permite la creación de aplicaciones ricas de Internet. Este tipo de aplicaciones se caracteriza por una alta interactividad con el usuario, comportándose como una aplicación de escritorio, sin necesi-

dad de instalar ningún software en cliente, solamente un navegador. El sistema de control supervisorio puede beneficiarse grandemente de estas características y utilizarlas para mantener informado constantemente al usuario de las variaciones en los valores de los parámetros objeto de supervisión. Aunque este marco de trabajo no es el único que permite la creación de este tipo de aplicaciones (los productos que implementan AJAX también lo hacen), es uno de los más maduros dentro de la categoría de código abierto.

En la capa de la lógica de negocios Java es el lenguaje mas favorecido. La especificación J2EE abarca gran parte de los servicios que se necesitan en esta capa, y continúa en evolución para incorporar nuevos y mejorar los existentes. Existen varios productos que implementan la especificación J2EE total o parcialmente (servidores de aplicaciones), algunos son gratuitos como JBoss y GlassFish y otros como Oracle AS, Bea WebLogic e IBM WebSphere, son comerciales.

Además de los servidores de aplicaciones existe un marco de trabajo que constituye una alternativa en costo y simplicidad a éstos. Spring [16] es un marco de trabajo ligero, no invasivo, de código abierto, basado en patrones de diseño, que simplifica la creación de la capa de lógica de negocios tanto para aplicaciones WEB como de escritorio. Inicialmente solo estaba disponible para el lenguaje Java. En la actualidad el producto se encuentra disponible también para la tecnología .NET. Este fue el marco de trabajo seleccionado para la capa de lógica de negocios.

La capa de persistencia también cuenta con productos que simplifican el trabajo con la misma, entre los que se encuentran Hibernate, IBatis, EJB3.0, entre otros. El más popular y utilizado de éstos es Hibernate [17]. Este producto suministra un potente servicio de consultas y persistencia objeto relacional de alto rendimiento. Permite el desarrollo de clases persistentes, incluyendo asociación, herencia, polimorfismo, composición y colecciones. La consultas se expresan utilizando HQL (Hibernate Query Language), SQL nativo, o haciendo uso de las API Criteria, Query y Example. Aunque inicialmente sólo estaba disponible para el lenguaje Java, en la actualidad existe una versión para la tecnología .NET. Este fue el marco de trabajo seleccionado para esta capa.

En el momento del comienzo de este proyecto, la única tecnología realmente multiplataforma[1], de código abierto, con amplio soporte de los entornos de desarrollo integrado, y con una amplia gama de marcos de trabajo que simplifican el desarrollo de las diferentes capas de la aplicación era Java; por lo que fue la elegida para el desarrollo del mismo.

[1] Aunque se conoce de la existencia del proyecto MONO, que pretende portar el marco de trabajo de .NET hacia Linux, todavía no se encontraba en una etapa que permitiera el desarrollo de un proyecto como este.
PHP es otra alternativa en cuanto a la multiplataforma y a los requerimientos de licencia, pero no contaba con la cantidad de marcos y bilbiotecas de trabajo que disponía Java, que permitía un mayor nivel de reusabilidad.

CAPÍTULO III. DISEÑO DE UN SISTEMA DE CONTROL SUPERVISORIO PARA DISPOSITIVOS GESTIONABLES POR HTTP.

El diseño del software que ejecuta el control supervisorio de dispositivos de control con soporte TCP/IP y HTTP estuvo guiado por la metodología de desarrollo de software Rational Unified Process (RUP) [19].

3.1 EL PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE

El Proceso Unificado es el fruto de tres décadas de desarrollo y uso práctico de diversas metodologías para el desarrollo de software. Su evolución ha estado marcada por la influencia de muchas fuentes, cada una de las cuales aportó sus mejores prácticas, muchas de las cuales se encuentran presente en el proceso actual. La figura No. 16 muestra la evolución de esta metodología.

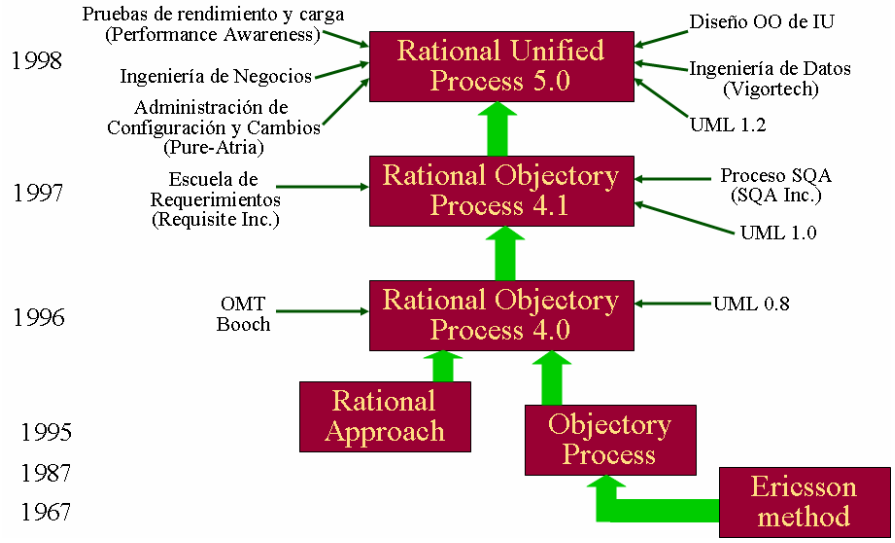


Figura No. 16. Evolución del Proceso Unificado de Racional

3.1.1 CARACTERÍSTICAS FUNDAMENTALES DEL PROCESO UNIFICADO DE SOFTWARE

Un proceso de desarrollo de software es el conjunto de actividades que permiten transformar los requisitos de los clientes en un sistema de software [19]. El Proceso Unificado de Software

(RUP) no es solamente un proceso de software, sino que constituye un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos.

Aunque está basado en componentes y utiliza UML, los aspectos definitorios de RUP pueden expresarse como:

- ✓ Dirigido por casos de uso
- ✓ Centrado en la arquitectura
- ✓ Iterativo e Incremental

3.1.1.1. EL PROCESO UNIFICADO ESTÁ GUIADO POR CASOS DE USO

El objetivo de todo sistema de software es servir a sus usuarios, por tanto hay que conocer con exactitud lo que los usuarios necesitan y desean. Un usuario puede ser una persona o sistema externo que interactúa con el sistema de software que se está desarrollando. La interacción de un usuario con el sistema se denomina caso de uso, y éstos representan los requisitos funcionales del sistema, o sea, aquellas cosas que el sistema debe hacer y que producen un resultado de valor para los usuarios. La agrupación de casos de usos se conoce como modelo de casos de uso y describe la funcionalidad total del sistema. En el modelo de casos de uso se debe encontrar respuesta a la pregunta "¿Qué debe hacer el sistema para cada usuario?".

Dirigido por casos de uso significa que el proceso de desarrollo avanza a través de una serie de flujos de trabajo que parten de los casos de uso. Éstos se desarrollan a la vez que la arquitectura del sistema, y ésta influye en la selección de los casos de uso. Ambos maduran a lo largo del proceso de desarrollo de software.

3.1.1.2 EL PROCESO UNIFICADO ESTÁ CENTRADO EN LA ARQUITECTURA

El concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema, es una vista del diseño completo con las características más importantes resaltadas, dejando a un lado los detalles. El valor de la arquitectura depende de las personas involucradas en su realización, debido a que en este proceso intervienen elementos que se adquieren en la mayoría de los casos a través de la experiencia.

Cada producto tiene una función y una forma. Ambas deben complementarse para obtener un producto de éxito. En términos de un producto software la función está relacionada con los casos de uso y la forma con la arquitectura [19], ambas deben evolucionar en paralelo a lo largo del ciclo de vida del proyecto.

3.1.1.3 El Proceso Unificado es iterativo e incremental

Para llevar a cabo la ejecución de un proyecto de software (el cual involucra muchos esfuerzos y puede durar desde pocos meses hasta años) resulta práctico dividirlo en micro proyectos para su mejor gestión. Cada macroproyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en el flujo de trabajo y los incrementos al crecimiento del producto. Para alcanzar máxima efectividad las iteraciones deben ser controladas, es decir, deben seleccionarse y ejecutarse de forma planificada. Un proceso iterativo controlado permite:

- ✓ Reducir el costo
- ✓ Reducir el riesgo de incumplir con los plazos de entrega
- ✓ Acelera el ritmo de desarrollo
- ✓ Adaptarse a los requisitos cambiantes

El equipo de proyecto intentará seleccionar solamente las iteraciones necesarias para lograr el objetivo trazado y secuenciarlas en un orden lógico.

3.1.2 Ciclo de vida del Proceso unificado

El Proceso Unificado se ejecuta a lo largo de una serie de ciclos, como se muestra en la figura No. 17. Éstos constituyen la vida del proceso y se desarrollan a lo largo del tiempo, cada ciclo concluye con una versión del producto para los clientes y consta de cuatro fases, las cuales concluyen con un hito bien definido que permite tomar decisiones: inicio, elaboración, construcción, transición.

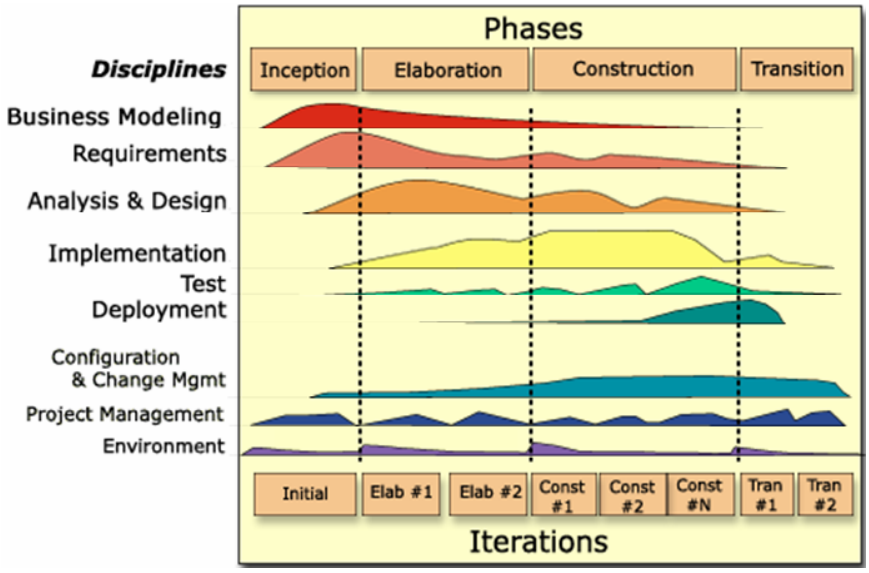


Figura No. 17. Ciclo de vida del Proceso Unificado

Una fase en el ciclo es el intervalo de tiempo entre dos hitos importantes del proceso durante el que se cumple un conjunto bien definido de objetivos, se completan partes del sistema y se toman decisiones sobre si pasar o no a la siguiente fase. Dentro de cada fase hay varias iteraciones.

Una iteración representa un ciclo de desarrollo completo, desde la captura de requisitos en el análisis hasta la implementación y pruebas, que produce como resultado la entrega al cliente o la salida al mercado de un proyecto ejecutable.

Cada iteración pasa a través de varios flujos de trabajo del proceso, aunque con un énfasis diferente en cada uno de ellos, dependiendo de la fase en que se encuentre:

1. Iniciación, se establece la oportunidad y alcance del proyecto.
2. Elaboración se analiza el dominio del problema se establece una arquitectura sólida y el interés se orienta hacia el análisis y el diseño.
3. Construcción, la actividad central es la implementación.
4. Transición se centra en despliegue.

En el anexo No. IV se muestran los modelos producidos en cada uno de los flujos de trabajo, y en el anexo No. V se relacionan los artefactos incluidos en cada modelo.

3.2 Diseño del sistema de control supervisorio utilizando el Proceso Unificado de Software

El diseño del sistema de control supervisorio estuvo regido por la metodología Proceso Unificado de Rational, específicamente estuvieron involucrados los flujos Requisitos y Análisis y Diseño. El desarrollo del sistema estuvo soportado por al paquete de software Rational Rose 2003, el cual incorpora herramientas para la creación de los diferentes artefactos resultado de las diferentes etapas del desarrollo.

3.2.1 Modelado de Negocio

En esta etapa se deben identificar y modelar los procesos de negocios objeto de implementación por el producto de software bajo desarrollo, así como la identificación de los actores de negocios y los casos de uso de negocio

En el caso en particular del sistema supervisorio, no existe un proceso de negocio como tal, no existe una contraparte no automatizada de la problemática que debe resolver el producto software, mas bien productos en el mercado con alcances y objetivos similares, pero en la mayoría de los casos sobredimensionados para la problemática a resolver y con un costo muy elevado.

3.2.2 Gestión de Requisitos

El propósito fundamental de esta etapa consiste en guiar al desarrollo del software por el camino correcto, mediante la correcta identificación y descripción de los requisitos del sistema, o sea, todas aquellas condiciones o capacidades que el sistema debe cumplir y que aportan resultados de valor a los clientes. Este proceso debe ser lo suficientemente bueno como para que los clientes y desarrolladores lleguen a un acuerdo acerca qué debe hacer y qué no debe hacer el sistema bajo desarrollo. El cliente debe ser capaz de leer y comprender el resultado de esta etapa, para lo cual se debe emplear un lenguaje lo suficientemente claro para que esto se cumpla.

En la identificación y captura de requisitos del sistema supervisorio se transitaron por tres etapas:

1. Enumerar los requisitos candidatos (listas de características)
2. Captura de los requisitos funcionales
3. Captura de los requisitos no funcionales

En esta etapa se realizaron actividades participativas en las que participaron el equipo de desarrollo, un conjunto de expertos en la explotación e instalación de sistemas de este tipo y un grupo de clientes actuales de sistemas similares al que se desea desarrollar y con amplia experiencia en la utilización de los mismos como para realizar un análisis crítico de éstos. La lista de los requisitos candidatos se muestra en el anexo No. VI

Los requisitos funcionales se reflejaron a través del modelo de casos de uso, de cada caso de uso se realizó su descripción identificando actores, flujos de actividades, poscondiciones, precondiciones, puntos de extensión, etc.

Para su mejor organización el modelo de casos de uso se estructuró en paquetes, los que se muestran en el anexo No. VII.

El trabajo con la lista de requisitos candidatos produjo el modelo de casos de uso de la aplicación. En el anexo VIII se muestran los diagramas de casos de uso y en los anexos del No. IX al No. XIX se recogen las especificaciones de los mismos.

3.2.3 Análisis

El flujo de trabajo del proceso de análisis trajo como resultado la identificación de las clases de interfaz, de control y de entidad, así como su interacción para llevar a cabo los diferentes casos de uso, identificados en la etapa de gestión de requisitos.

En el anexo No. XX se muestran los artefactos producidos en esta etapa.

3.2.4 Diseño

El núcleo del sistema lo compone la unidad de control. Una unidad de control está caracterizada por su dirección IP y tiene asociada además el número de puerto TCP por el que escucha el servidor Web embebido, incluye también información acerca de la ubicación geográfica de la misma, descripción y disponibilidad. Contiene además un conjunto de parámetros que pueden corresponderse con las entradas físicas conectadas a ella o no. El usuario podrá per-

sonalizar la apariencia de la unidad de control en el subsistema de visualización indicando la imagen que asociará a ésta.

Los parámetros están caracterizados por la dirección IP de la unidad de control a la que pertenecen y por los URL que tienen asignado dentro de la misma. Para cada parámetro se define un URL para obtener información de la unidad de control (lectura) y otro para enviar información hacia la unidad de control (escritura). En caso que un parámetro esté asociado a una entrada física, se podrá conocer el valor de esa entrada utilizando el URL de lectura.

Un parámetro no necesariamente deberá estar asociado a una entrada física en la unidad de control. Puede definirse un parámetro cuyo objetivo sea solamente el de enviar información, a través del URL de escritura. Ese parámetro puede ser utilizado para establecer los valores de variables que serán consultadas por el algoritmo de control que se ejecuta dentro del dispositivo. En este caso el parámetro no reporta interés alguno a los efectos de la medición y será excluido de este proceso.

Los parámetros además contienen información que permite conocer:

- ✓ Una descripción de la entrada física correspondiente en la unidad de control.
- ✓ El intervalo de medición, o sea, cada cuanto tiempo debe realizarse una medición. Los parámetros de una misma unidad de control pueden tener diferentes intervalos de medición.
- ✓ Si la medición se almacenará en la estructura de almacenamiento temporal (escribible).
- ✓ Si la medición se almacenará en la estructura de almacenamiento histórica (archivable).
- ✓ Si una medición de este parámetro genera una alarma si su valor se encuentra fuera de los umbrales permisibles.
- ✓ Si el parámetro se incluirá en el proceso de medición.
- ✓ Si en el momento de obtener el estado del parámetro se deberá consultar directamente a la unidad de control o se extraerá la información de la estructura de almacenamiento temporal.
- ✓ Los valores límites de los umbrales permisibles
    - o Bajo
    - o Medio
    - o Alto

Al igual que las unidades de control, la apariencia visual de un parámetro en el subsistema de visualización podrá ser personalizada por el usuario, indicando las imágenes que asociará a los diferentes estados del mismo.

Se disponen de tres tipos de parámetros, que representan los posibles valores a obtener de las entradas físicas conectadas a las unidades de control: lógicos, enteros y reales.

Una medición está caracterizada por el momento en que se realiza, el parámetro al que corresponde y el valor de la entrada asociada. Si el parámetro es escribible se almacenará en la estructura de almacenamiento temporal, y si es archivable se almacenará en la estructura de almacenamiento histórica. El estado de la medición se conoce a través del proceso de normalización de la misma, en el cuál se tienen en cuenta los valores límites de los umbrales definidos en el parámetro.

Una alarma es una medición cuyo valor esta por encima de límite superior del umbral alto o por debajo del límite inferior del umbral bajo. Si el parámetro ha sido configurado para que genere alarma, luego de una medición se determinará si ésta constituye una alarma o no, en cuyo caso se almacenará y se notificará la ocurrencia de este evento.

Existen también un conjunto de parámetros operacionales sobre los cuales se sustentan las decisiones tomadas en el proceso de medición:

- ✓ El umbral de preferencia para las mediciones cuyos valores corresponden con los límites frontera (valor frontera entre el umbral bajo/medio y entre el umbral medio/alto)
- ✓ Dirección de correo electrónico para notificar las alarmas
- ✓ Tiempo de vida de la estructura de almacenamiento temporal
- ✓ Tiempo de vida de la estructura de almacenamiento histórico
- ✓ Tiempo de vida de la estructura de almacenamiento de alarmas
- ✓ Cantidad de temporizadores de medición y valor de cada temporizador

La aplicación estará estructurada en tres subsistemas:

- ✓ Subsistema de Administración
- ✓ Subsistema Operacional
- ✓ Subsistema de Visualización

3.2.4.1. SUBSISTEMA DE ADMINISTRACIÓN

Este subsistema tiene a su cargo la configuración de los parámetros de operación y la administración de las unidades de control y usuarios del sistema, permite:

- ✓ Gestión de las unidades de control presentes en el sistema
- ✓ Gestión de los parámetros de las unidades de control presentes en el sistema
- ✓ Gestión de los usuarios que tendrán acceso al sistema
- ✓ Gestión de los parámetros operacionales del sistema

Este subsistema realiza los casos de uso CU001, CU002, CU003, CU004, CU010 y CU011 (ver anexo VIII)

3.2.4.1.1. Gestión de las unidades de control y de los parámetros de las mismas

La gestión de las unidades de control comienza con la detección de las unidades instaladas en un rango de direcciones IP o una dirección IP en particular, direcciones éstas que el usuario debe introducir en una interfase creada al efecto. Además de las direcciones IP se deberá indicar el puerto TCP por el que escucha el servidor de gestión embebido en la unidad de control.

El sistema intentará establecer una conexión TCP con cada unidad de control utilizando la dirección IP de ésta y el número de puerto. De lograr establecerse la conexión se almacenará la unidad de control en las estructuras de almacenamiento del sistema.

La unidad (o las unidades) detectada aparecerá en la interfase creada al efecto. Seleccionando una unidad en particular se podrá modificar la información relacionada con ésta (ver diagrama de clases en el anexo XXII). Además de la información relacionada con el funcionamiento de la unidad se podrá también modificar la información relacionada con la apariencia visual que tendrá la misma en el subsistema de visualización.

La interfase proveerá una opción para añadir parámetros a la unidad seleccionada. Mediante una interfase creada al efecto se podrá introducir la información de cada parámetro (ver diagrama de clases en el anexo XXII). También se podrá modificar la información relativa a la apariencia visual de los diferentes estados del parámetro en el subsistema de visualización.

3.2.4.1.2. Gestión de usuarios

La gestión de usuarios permitirá adicionar, modificar y eliminar los usuarios que tendrán acceso a los diferentes subsistemas. Mediante una interfase creada al efecto se mostrarán los usuarios presentes en el sistema. Al seleccionar un usuario se podrá modificar la información relacionada con éste o eliminar el mismo. De igual manera la interfase proveerá una opción para adicionar nuevos usuarios al sistema.

3.2.4.1.3. Gestión de los parámetros operacionales del sistema

Durante el funcionamiento del subsistema operacional se deben tomar decisiones basadas en las preferencias del propietario de la instalación a supervisar. La función de este módulo es suministrar una interfase mediante la cual se puedan gestionar estas preferencias.

3.2.4.2. SUBSISTEMA OPERACIONAL

Este subsistema es el encargado de realizar el monitoreo de las unidades de control, realiza las mediciones de los parámetros de las mismas y las almacena teniendo en cuenta las propiedades de los éstos. Está compuesto por dos módulos:

- ✓ Módulo de planificación
- ✓ Módulo de medición

El módulo de planificación es el encargado de planear la medición de los parámetros, recupera los parámetros medibles de las unidades activas presentes en el sistema, determina para cada parámetro el tiempo de medición y lo notifica al módulo de medición.

Por su parte el módulo de medición obtiene el valor de los parámetros medibles de las unidades activas, normaliza la medición, determina si constituye una alarma o no (en cuyo caso se notifica este evento), y se encarga de garantizar la persistencia de las mediciones. Todo esto teniendo en cuenta la configuración de cada parámetro.

El módulo operacional constituye el núcleo de la aplicación, no tiene interacción con el usuario, por lo que carece de interfaces gráficas. Debe correr permanentemente, como tarea de fondo en el PC. En la programación de este módulo se utilizarán las características multihilo del lenguaje de programación. Este subsistema realiza el caso de uso CU005

3.2.4.3. SUBSISTEMA DE VISUALIZACIÓN

El módulo de visualización muestra al usuario el estado de los diferentes parámetros de las unidades de control de forma gráfica (mímico) y de forma detallada (tabular). Igualmente muestra al usuario información acerca de las alarmas y el comportamiento de los parámetros en el tiempo.

La información general acerca del estado de un determinado dispositivo se mostrará en forma de mímico. En esta imagen se podrá observar el estado de los diferentes parámetros asociados al dispositivo en cuestión. La imagen asociada a cada dispositivo, así como las imágenes asociadas a los estados normalizados de cada parámetro, serán susceptibles de personalización por parte del usuario.

La selección de un mímico dará paso a la información tabular sobre el dispositivo. En esta vista se obtendrá mas detalles acerca del estado de los diferentes parámetros, se podrán conocer sus valores numéricos y será posible además seleccionar las mediciones de los mismos en un intervalo de tiempo. También será posible observar de forma gráfica el comportamiento de un parámetro en el tiempo.
Las alarmas por su parte se mostrarán en formato tabular con el mayor nivel de detalle posible.

Este subsistema realiza los casos de uso CU001, CU008, CU009, CU006 y CU007. La etapa de diseño trajo como resultado los artefactos mostrados en los anexos XXI al XXII.

3.4.4.4. SOBRE LA IMPLEMENTACIÓN

El sistema se implementará como una aplicación multicapa utilizando la tecnología Java y haciendo uso de patrones de diseño como mejor práctica en la ingeniería se software.

No existen restricciones en cuanto a la implementación de la capa de recursos. Se utilizará cualquier sistema de bases de datos relacionales, aunque se recomienda el uso de productos gratuitos, y de código abierto, en especial MySQL (http://www.mysql.org), por su probado rendimiento y disponibilidad en los sistemas operativos más utilizados

En la capa de integración se utilizará un marco de trabajo denominado Hibernate (http://www.hibernate,org), el cual permitirá realizar la correspondencia objeto relacional de las clases y manejar todos los aspectos relacionados con la persistencia de los objetos. Este producto adiciona una capa de abstracción de la aplicación de los posibles sistemas de almacenamientos (bases de datos) a utilizar, garantizando la portabilidad de la misma entre estos sistemas de almacenamiento.

La capa de lógica de negocios hará uso de los servicios del marco de trabajo Spring (http://www.springframework.org). Este producto es una alternativa ligera, totalmente gratuita a los servidores de aplicaciones comerciales, que facilita la creación de la capa de lógica de negocios y puede utilizarse en aplicaciones de escritorio y en aplicaciones WEB.

La capa de presentación utilizará la tecnología denominada WEB2, la cual es el resultado de la evolución de las aplicaciones WEB, y añade la interactividad característica de las aplicaciones de escritorio. En esta capa se empleará un marco de trabajo de código abierto denominado Open Laszlo (http://www.openlaszlo.org/), el cual se ejecuta en un contenedor de SERVLETS y JSP, y permite el desarrollo de aplicaciones WEB utilizando JavaScript y XML como lenguajes. La interacción hacia las capas inferiores de la aplicación se realiza a través de XML y la salida final que se envía al navegador está en formato Flash; en futuras versiones será también DHTML.

3.2.4.5 Sobre el despliegue

Las unidades de control deberán ser instaladas en una red de área local, bien sea empleando cualquier tecnología de la familia 802.3 o 802.11 de IEEE. Se recomienda que la red de las unidades de control se encuentre aislada del resto de las redes desde el punto de vista físico, a fin de evitar tráfico innecesario y accesos no deseados a ésta. Si los parámetros han sido configurados de forma tal que para la visualización su valor se tomará siempre de la última medición realizada, entonces los únicos subsistemas que necesitan acceder a la red de las unidades de control son el subsistema operacional y el de administración, como se muestra en la figura No. 18, de lo contrario, todos los subsistemas deberán tener acceso a la red de las unidades de control. El enrutador que se muestra en la figura solamente permitirá el acceso del subsistema de visualización, ubicado en la red de oficina, al subsistema de almacenamiento, que se encuentra en la red de control. En configuraciones más sencillas el enrutador puede

eliminarse convirtiendo al equipo donde corre el subsistema operacional en un “dual homed host”.

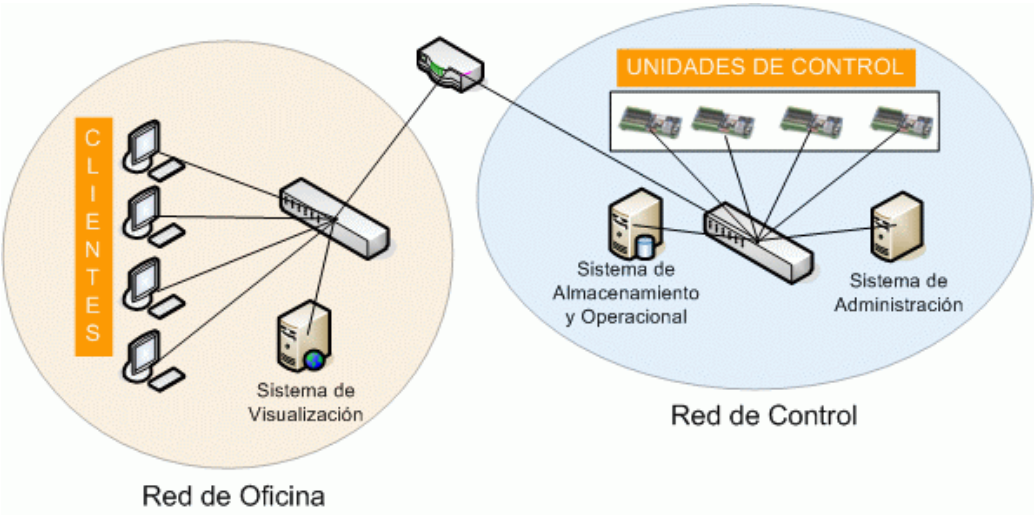


Figura No. 18. Topología de las redes de oficina y control

Para el despliegue de la aplicación se proponen dos alternativas, la selección de cada una de ellas depende de los recursos disponibles y del rendimiento deseado.

La primera alternativa es (ver anexo XXIII (A)) es la más sencilla de implementar, pero el costo dependerá del rendimiento esperado, ya que al estar todos los subsistemas y recursos concentrados en un solo equipo, para lograr un rendimiento aceptable las prestaciones del equipo deben ser altas.

La segunda alternativa (ver anexo XIII (B)) involucra mas de un equipo. En este caso, las prestaciones de los equipos pueden ser menores a las del equipo de la alternativa No. 1, aunque aumenta la cantidad de éstos. También en este caso aumenta el tráfico entre los equipos donde se encuentran los diferentes subsistemas.

CONCLUSIONES

En el presente trabajo se ha abordado el diseño de un sistema de control supervisorio para dispositivos de control gestionables a través de HTTP, utilizando la tecnología Web. Se han elaborado las especificaciones técnicas del software, se ha diseñado la arquitectura y se han elaborado los artefactos correspondientes a las etapas de gestión de requisitos, análisis y diseño según la metodología de desarrollo de software RUP. El sistema diseñado puede ser utilizado también con dispositivos que no sean gestionables directamente a través de HTTP mediante el uso de pasarelas.

La metodología RUP, debido a la gran cantidad de artefactos que genera y la propia estructuración del ciclo de desarrollo e software, resulta un tanto incómoda cuando es necesario realizar cambios continuos en la gestión de requisitos. Además, resulta lenta en proyectos de poca envergadura y equipos de trabajos pequeños.

Las investigaciones realizadas durante la ejecución de este trabajo permitieron constatar que en el área de la automatización se aprecia una gran cantidad de productos gestionables a través de TCP/IP y HTTP, lo cual pone de manifiesto la versatilidad de estos protocolos. Por su parte, Los sistemas SCADA comerciales, además de dar soporte a las formas tradicionales de comunicación con los dispositivos de automatización, han comenzado a incorporar TCP/IP y HTTP, para dar respuesta al crecimiento de productos gestionables por esta vía.

En la esfera de las tecnologías de desarrollo de aplicaciones Web la tecnología Java aparece como la más beneficiada debido a la gran cantidad de productos de código abierto disponibles, las cuales abarcan todas las capas de una aplicación Web multicapa.

RECOMENDACIONES

1. Continuar el trabajo iniciado ejecutando las fases de implementación y pruebas.
2. Crear pasarelas para los dispositivos de automatización más utilizados en el país (Siemens, LG, Omron)
3. Explorar el uso de metodologías ágiles de desarrollo de software.

BIBLIOGRAFÍA

1. Tanenbaum Andrew. Computer Networks, 4 Edition. …..
2. J. Postel. Internet Protocol. http://www.ietf.org/
3. J. Postel. Internet Control Message Protocol. http://www.ietf.org/
4. J. Postel. Transmission Control Protocol. http://www.ietf.org/
5. J. Reynolds, J. Postel. Assigned Numbers. http://www.ietf.org/
6. Dalal, Y. and C. Sunshine. Connection Management in Transport Protocols. Computer Networks, Vol. 2, No. 6, pp. 454-473 Diciembre 1978.
7. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. Hypertext Transfer Protocol HTTP/1.1. http://www.ietf.org/
8. T. Berners-Lee, R. Fielding, H. Frystyk. Hypertext Transfer Protocol HTTP/1.0. http://www.ietf.org/
9. N. Freed, N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. http://www.ietf.org/
10. T. Berners-Lee, L. Masinter, M. McCahill. Uniform Resource Locators (URL). . http://www.ietf.org/
11. Mockapetris, P., "Domain Names - Concepts and Facilities. http://www.ietf.org/
12. Core servlets and JSP. Prentice Hall y Sun Microsystems, http://www.coreservlets.com
13. What is a SCADA? Axel Daneels**,** Wayne Salter http://ref.web.cern.ch/ref/CERN/CNL/2000/003/scada/Pr
14. The OPC Foundation. http://www.opcfoundation.org/
15. Mann, Kito D. Java Server Faces in Action. Manning Publications Co. 20005
16. Walls Craig, Breidenbach Ryan. Spring in Action. Manning Publications Co. 20005
17. Bauer Christian, King Gavin. Hibernate in Action. Manning Publications Co. 20005
18. http://es.wikipedia.org/
19. Jacobson Ivar, Booch Grady, Rumbaugh James. El proceso unificado de desarrollo de software. Addison Wesley, 1999.
20. ¿Java, un lenguaje mas?, Revista Telemática, Año I, No. 42
21. The Java Servlet API White Paper. http://java.sun.com/products/servlet/whitepaper.html

GLOSARIO DE TÉRMINOS

PLC

Dispositivos electrónicos muy usados en Automatización Industrial. Fue creado para sustituir los circuitos basados en relevadores empleados en el control de maquinarias y procesos.

RTU

Un RTU, o la Unidad Terminal Remota es un dispositivo que une los objetos en el mundo físico a un sistema de mando distribuido o sistema SCADA transmitiendo los datos al sistema y/o alterando el estado de los objetos conectados basado en mensajes recibidos del sistema.

RS232

RS-232 (también conocido como EIA RS-232C) es una interfaz que designa una norma para el intercambio serie de datos binarios entre un equipo terminal de datos (DTE) y un equipo de terminación del circuito de datos (DCE).

RS485

EIA-485 (anteriormente RS-485 o RS485) es una especificación de la capa física del modelo OSI de una conexión serie multi punto, de dos cables en modo semi duplex.

Ethernet

Ethernet es el nombre de una tecnología de redes de computadoras de área local basada en tramas de datos. Define las características de cableado, señalización de nivel físico y los formatos de trama del nivel de enlace de datos del modelo OSI bajo el estándar IEEE 802.3.

Host

Una computadora conectada a una red de computadoras a la cual se le ha asignado un identifi-

cador para su comunicación con otros hosts.

Modelo OSI

El modelo de referencia de Interconexión de Sistemas Abiertos (OSI, Open System Interconection) lanzado en 1984 fue el modelo de red descriptivo creado por ISO. Proporcionó a los fabricantes un conjunto de estándares que aseguraron una mayor compatibilidad e interoperabilidad entre los distintos tipos de tecnología de red producidos por las empresas a nivel mundial.

RFC

Acrónimo en ingñés de “Request For Comments”. Conjunto de notas técnicas y organizativas donde se describen los estándares o recomendaciones de Internet (originalmente ARPANET), comenzado en 1969.

MIME

MIME (Multipurpose Internet Mail Extensions, Extensiones de Correo Internet Multipropósito), son una serie de convenciones o especificaciones dirigidas a que se puedan intercambiar a través de Internet todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario. Una parte importante del MIME está dedicada a mejorar las posibilidades de transferencia de texto en distintos idiomas y alfabetos.

XML

XML (sigla en inglés de eXtensible Markup Language, lenguaje de marcado extensible) es un lenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (cómo HTML es un lenguaje definido por SGML). No es realmente un lenguaje en particu-

	lar, sino una manera de definir lenguajes para diferentes necesidades.
Hipermedia	Hipermedia se le designa al conjunto de métodos, o procedimiento para escribir, diseñar, o componer contenidos que tengan texto, video, audio, mapas u otros medios, y que además tenga la posibilidad de interactuar con los usuarios.
Applets	Un applet es un componente de software que corre en el contexto de otro programa, por ejemplo un navegador web. El applet debe correr en un contenedor, que es proporcionado por un programa anfitrión, no puede correr de manera independiente y tiene privilegios de seguridad restringidos.
COM	Component Object Model (COM) es una plataforma de Microsoft para componentes de software introducida por Microsoft en 1993. Esta plataforma es utilizada para permitir la comunicación entre procesos y la creación dinámica de objetos, en cualquier leguaje de programación que soporte dicha tecnología.
DCOM	El Modelo de Objetos de Componentes Distribuidos (Distributed Component Object Model o DCOM) es una tecnología propietaria de Microsoft para desarrollar componentes software distribuidos sobre varios ordenadores y que se comunican entre sí. Extiende el modelo COM de Microsoft y proporciona el sustrato de comunicación entre la infraestrcutura del servidor de aplicaciones COM+ de Microsoft.Ha sido abandonada en favor del framework .NET.

GTK

GTK+ es un grupo importante de bibliotecas o rutinas para desarrollar interfaces gráficas de usuario (GUI) para principalmente los entornos gráficos GNOME, XFCE y ROX de sistemas Linux. Es software libre (bajo la licencia LGPL), multiplataforma y parte importante del proyecto GNU.

Bytecode

El bytecode es un código intermedio más abstracto que el código máquina. Habitualmente se le trata como a un fichero binario que contiene un programa ejecutable similar a un módulo objeto, que es un fichero binario que contiene código máquina producido por el compilador.

El bytecode recibe su nombre porque generalmente cada código de operación tiene una longitud de un byte, si bien la longitud del código de las instrucciones varía. Cada instrucción tiene un código de operación entre 0 y 255 seguido de parámetros tales como los registros o las direcciones de memoria.

UML

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aún cuando todavía no es un estándar oficial, está apoyado en gran manera por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

ANEXOS

ANEXO I TIPOS MIME MAS UTILIZADOS Y SU SIGNIFICADO

Tipo MIME	Significado
application/msword	Documento de MS Word
application/octet-stream	Dato binario
application/pdf	Documento en formato PDF
application/postscript	Documento en formato PostScript
application/vnd.lotus-notes	Documento de Lotus Notes
application/vnd.ms-excel	Hoja de cálculo Excel
application/vnd.ms-powerpoint	Documento de Power Point
application/x-gzip	Archivo comprimido con Gzip
application/x-java-archive	Archivo Jar
application/x-java-serialized-object	Objeto serializado Java
application/x-java-vm	Archivo .class de Java
application/zip	Archivo comprimido con Zip
audio/basic	Archivo de audio .au o .snd
audio/x-aiff	Archivo de audio en formato AIFF
audio/x-wav	Archivo de audio en formato WAV
audio/midi	Archivo de audio en formato MIDI
text/css	Hoja de estilo en cascada
text/html	Documento HTML
text/plain	Texto plano
image/gif	Imagen en formato GIF
image/jpeg	Imagen en formato JPEG
image/png	Imagen en formato PNG
image/tiff	Imagen en formato TIFF
image/x-xbitmap	Imagen de mapa de bits de X Windows
video/mpeg	Archivo de video en formato MPEG
video/quicktime	Archivo de video de QuickTime

ANEXO II. CÓDIGOS DE ESTADOS MÁS UTILIZADOS

100: Si el servidor recibe un encabezado EXPECT con el valor “100-continue” significa que el cliente desea enviar un documento anexado en otra solicitud. En ese caso el servidor responderá con un código 100 si acepta o 417 si no acepta.

200: Indica que la solicitud ha sido procesada exitosamente.

201: El servidor ha creado un nuevo recurso en respuesta a una solicitud, el encabezado LOCATION indica su URL.

202: La solicitud ha sido aceptada, pero el procesamiento de la misma no ha concluido todavía.

203: El recurso se ha transferido con éxito, pero algunos encabezados de la respuesta pueden estar desactualizados debido a que se transfirió una copia y no el recurso original.

204: El cliente debe seguir visualizando el recurso anterior ya que el nuevo recurso no esta disponible.

205: Indica que no hay un nuevo recurso a transferir, pero el cliente está forzado a refrescar el recurso utilizando su copia local.

206: Se ha transferido una porción del documento.

300: El recurso solicitado puede ser obtenido en diferentes ubicaciones, listadas en la respuesta.

301: El recurso ya no se encuentra en la ubicación indicada por el URL, el encabezado LOCATION indica su nueva ubicación.

302: Similar a 301, pero el URL indicado en el encabezado LOCATION debe ser interpretado como su ubicación temporal, no permanente.

303: Similar a 301 y 302, pero si el método de la solicitud fue POST, el recurso indicado en el encabezado LOCATION debe ser recuperado con el método GET.

304: La versión almacenada localmente en el cliente no difiere de la que se encuentra en el servidor y el cliente debe utilizar la primera.

305: El recurso debe ser recuperado a través del Proxy cuyo URL se indica en el encabezado LOCATION.

400: Error de sintaxis en la solicitud del cliente.

401: El cliente trata de acceder a un recurso protegido por contraseña y no ha suministrado la información esperada en el campo AUTHORIZATION. La respuesta incluye un encabezado WWW-AUTHENTICATE.

403: El servidor rechaza la solicitud.

404: El recurso no se encuentra disponible en el URL suministrado por razones desconocidas.

405: El método indicado en la solicitud no está disponible para el recurso.

406: El recurso es de un tipo MIME incompatible con los tipos que acepta el cliente

407: Similar al 407 pero aplicable a los proxies.

408 El cliente ha consumido demasiado tiempo enviando la solicitud.

410: El recurso no existe en el servidor, a diferencia del 404, la causa es que el recurso no estará disponible de forma permanente.

411: El cliente debe suministrar el encabezado CONTENT-LENGTH, usualmente asociado a solicitudes con el método POST.

416: El cliente ha solicitado una porción del documento que no puede ser enviada por el servidor.

500: Indica que la entidad encargada de procesar la solicitud ha abortado o ha generado encabezados en un formato o sintaxis no reconocida.

501: El servidor no implementa la funcionalidad solicitada por el cliente.

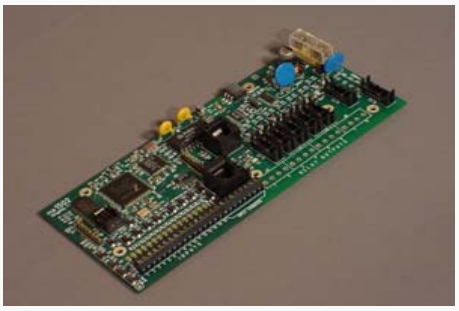
503: El servidor no puede responder debido a mantenimientos o sobrecarga.

505: El servidor no soporta la versión del protocolo indicada por el cliente.

ANEXO III DISPOSITIVO DE CONTROL CON SOPORTE TCP/IP Y HTTP.

RINGDALE
Connecting people and information

Search



Network and Web-Managed Programmable Logic Controller PLC-1608

Network and Web-Managed Programmable Logic Controller (PLC-1608) with 16 Inputs and 8 Outputs

This unique PLC allows up to 16 inputs to be evaluated and up to 8 relay isolated outputs to be controlled.

The device may be connected directly to a 10baseT or 100baseTx Ethernet LAN. A built in website allows for web-managed configuration and status display of the controller, therefore no front panel is required.

• Optional A/D and D/A board allowing for 8 Analog Input and 8 Analog Output channels to be evaluated or be controlled.
• Easy to understand programming language.

The controller is ideal in replacing older TTL type controllers such as those used to control half automatic screen printing equipment, security access curtains, punch presses, sheers and press breaks.

It is also ideal for remote control of motors and fans for factory or environmental monitoring applications.

The controller has a power supply and battery charger included, which can run the controller fully functioning for up to 8 days.

A built in timer and clock, which can syncronize with a timeserver or be connected to Ringdale's Satellite Time Server, keeps accurate time and can spawn actions at pre-set times.

The Ringdale PLC-1608 is ideal for solving small automation and robotics problems. It also integrtes easily into larger industrial applications. Communication between devices through the network gives the device unparalleled ability and flexibility.

Link to Applications:

Pressure/Dust Controlled Paint Booth

Half Automatic DEK Screen-Printer controller replacement

Pricing:

760.00
UK£ 475.00
US$ 736.25

Key Features:

- Built-in Ethernet 10/100baseTx network connection
- Built in Web-Server for Status and programming
- Easy to understand programming language
- Low power consumption makes it ideal for remote applications
- Flash programmable (retains program through long power failures)
- Programmable Logic Controller (PLC) with 16 inputs and 8 outputs
- Analog 8 port Input and 8 port Output option board is available
- Sends e-mail alerts directly from the device

Technical Specifications:

ANEXO NO. IV. MODELOS PRESENTES EN LOS DIFERENTES FLUJOS DE TRABAJO

	Modelado de Negocios	Requisitos	Análisis	Diseño	Implementación	Prueba	Despliegue
Modelo de Negocios	X						
Modelo de Dominio	X	X					
Modelo de Casos de Uso		X					
Modelo de Análisis			X				
Modelo de Diseño				X			
Modelo de Procesos				X			
Modelo de Despliegue				X			X
Modelo de Implementación					X		X
Modelo de Pruebas						X	X

ANEXO V. ARTEFACTOS PRESENTES EN LOS DIFERENTES MODELOS

Modelos	Negocio		Dominio		Casos de Uso		Análisis		Diseño		Procesos		Despliegue		Implemen-tación		Prueba	
Diagramas	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**
Casos de Uso	X				X												X	
Secuencia		X		X		X		X		X		X				X		X
Colaboración		X		X		X		X		X		X				X		X
Clases de Análisis							X											
Objetos de Análisis							X											
Clases de Di-seño									X		X							
Objetos de Diseño									X		X							
Estados						X		X		X		X		X		X		
Actividades								X		X		X		X		X		
Componentes															X			
Despliegue													X					

Leyenda:

E: Estáticos

D: Dinámicos

ANEXO VI LISTA DE REQUISITOS CANDIDATOS

Campo de aplicación

1. Debe permitir la supervisión de unidades de control con soporte TCP/IP y HTTP
2. Debe permitir supervisar unidades de control tradicionales

Unidades de Control

1. Adición de unidades de control de forma manual
2. Detección de unidades de control existentes en un rango de direcciones IP
3. Modificación de los datos de las unidades de control
4. Eliminación de unidades de control
5. Adición/Modificación/Eliminación de parámetros de las unidades de control de forma manual.

Configuración

1. Modificación de los parámetros de comportamiento del sistema
2. Almacenamiento de la configuración

Visualización

1. Visualización de forma gráfica del estado de los parámetros de las unidades de control (mímico).
2. Visualización de forma tabular del estado y el valor de un parámetro en particular de una unidad de control en un período.
3. Visualización de forma tabular de las alarmas ocurridas en un período
4. Visualización de la cronología de alarmas ocurridas en una unidad o grupo de unidades de control en el tiempo
5. Visualización de forma gráfica del comportamiento histórico de un parámetro en el tiempo
6. Visualización de forma tabular de todos los parámetros de una unidad de control en un período

Medición

1. Medición automatizada de los parámetros
2. Medición manual de los parámetros

Seguridad

1. Acceso basado en roles y controlado por usuarios a las diferentes funcionalidades del sistema

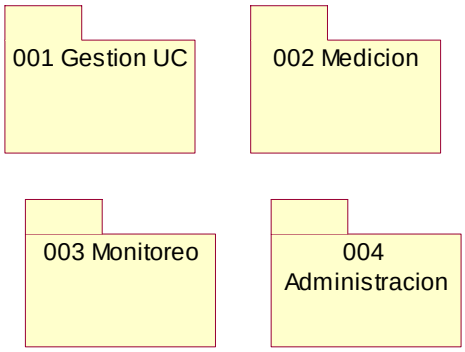
Almacenamiento

1. Almacenamiento de datos en estructuras temporales
2. Almacenamiento de datos en registros históricos

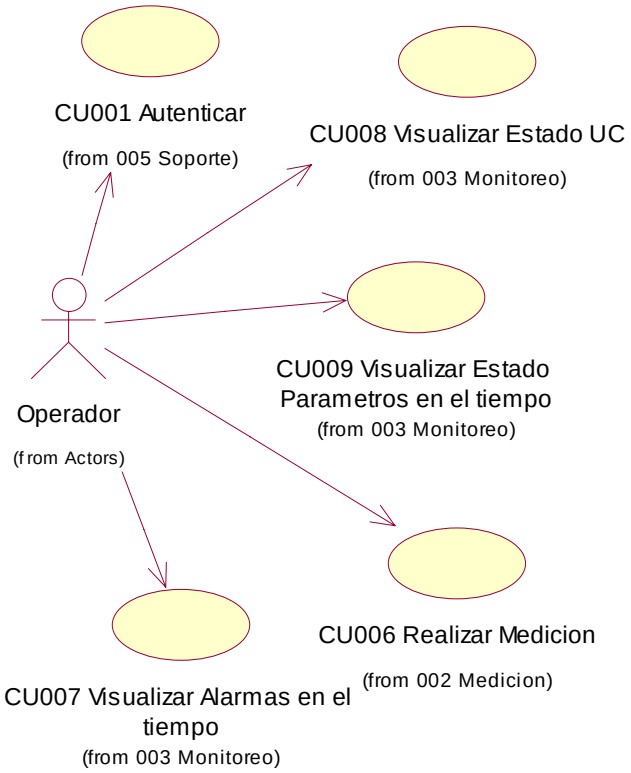
Tecnologías

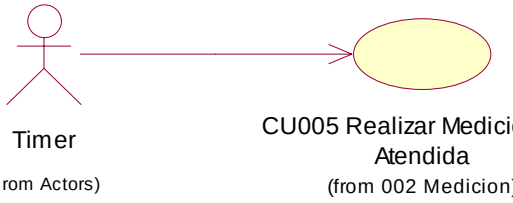
1. Multiplataforma (Independencia del sistema operativo)
2. Open Source
3. Plataforma Web

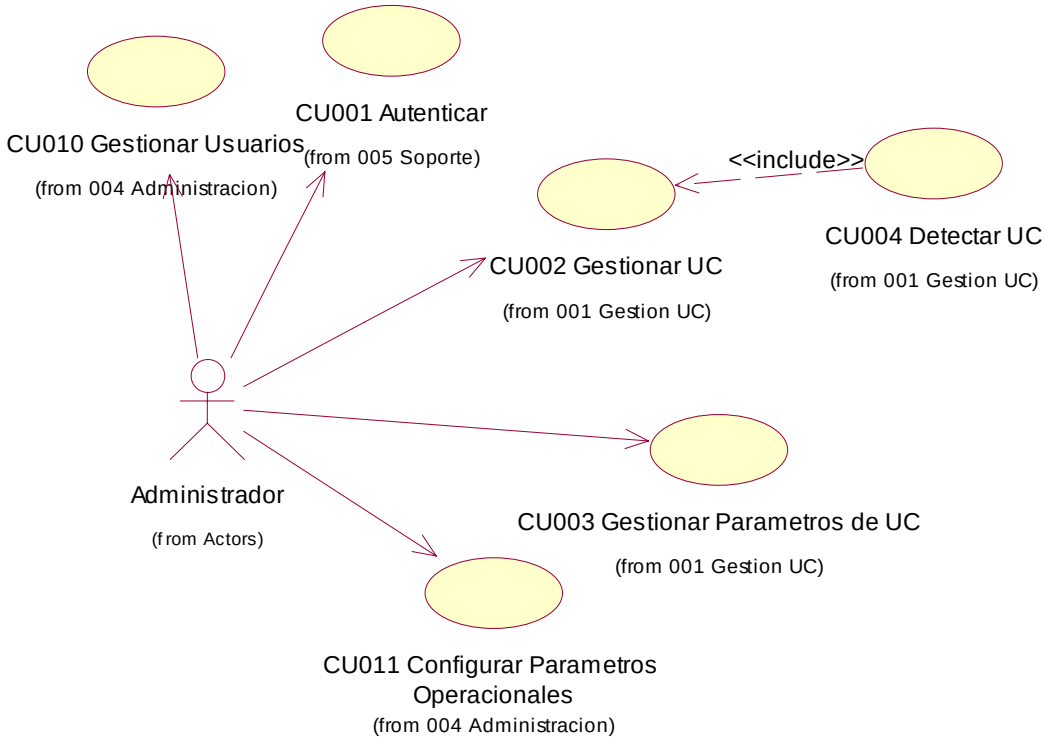
ANEXO VII. ESTRUCTURACIÓN DEL MODELO DE CASOS DE USO



ANEXO VIII. DIAGRAMAS DE CASOS DE USO






CU010 Gestionar Usuarios
(from 004 Administracion)
CU001 Autenticar
(from 005 Soporte)
<<include>>
CU004 Detectar UC
(from 001 Gestion UC)
CU002 Gestionar UC
(from 001 Gestion UC)
Administrador
(from Actors)
CU003 Gestionar Parametros de UC
(from 001 Gestion UC)
CU011 Configurar Parametros Operacionales
(from 004 Administracion)

ANEXO IX. CASO DE USO AUTENTICAR

1.- Nombre del Caso de Uso: CU001 Autenticar

2.- Descripción

Este caso de uso verifica que la información del cliente que trata de acceder al sistema es correcta.

3.- Precondiciones

El usuario ha accedido a la pantalla principal del sistema.

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el usuario accede a la pantalla principal del sistema.
2. El usuario introduce su identificador de usuario y su contraseña.
3. El sistema verifica que la información introducida por el usuario sea correcta y le muestra la pantalla donde aparecen las opciones del sistema
4. El caso de uso termina

Flujos Alternativos

Flujo Alternativo No. 1

1. En el punto 3 del flujo básico, si la información introducida por el usuario no es correcta el sistema le mostrará la pantalla inicial nuevamente.

5.- Poscondiciones

El usuario está listo para utilizar las opciones del sistema

ANEXO X. CASO DE USO GESTIONAR UC

1.- Nombre del Caso de Uso: CU002 Gestionar UC

2.- Descripción

Mediante este caso de uso el actor podrá gestionar la información asociada con las unidades de control.

3.- Precondiciones

El usuario se ha autenticado satisfactoriamente en el sistema

4.- Flujo de Eventos

Flujo Básico

5. El caso de uso comienza cuando el usuario ha accedido a la opción de Gestionar Unidades de Control
6. El sistema muestra las opciones de “Adicionar Unidades de Control”, “Modificar Unidades de Control” y ”Eliminar Unidades de Control”
7. El usuario selecciona la opción de “Adicionar Unidades de Control” (Flujo Alternativo No. 1, Flujo Alternativo No. 2)
8. El sistema muestra la pantalla para que el usuario introduzca los datos de la unidad de control
9. El usuario introduce los datos y guarda la información.
10. El sistema intentará detectar la presencia de una unidad de control utilizando los valores introducidos por el usuario a través del caso de uso CU004 Detectar UC.
11. El caso de uso termina cuando el usuario ha añadido, modificado o eliminado unidades de control.

Flujos Alternativos

Flujo Alternativo No. 1

1. En el paso 3 si el usuario selecciona la opción “Modificar Unidades de Control” deberá primero seleccionar de un listado la unidad de control que sesea modificar.
2. El sistema muestra una pantalla con la información de la unidad de control seleccionada para que el usuario haga modificaciones
3. El usuario realiza las modificaciones y guarda los cambios

Flujo alternativo No. 2

1. En el paso 3 si el usuario selecciona la opción “Eliminar Unidades de Control” deberá seleccionar primero de un listado la unidad de control que desea eliminar
2. El sistema le pedirá que confirme la acción
3. Si el usuario confirma la acción el sistema eliminará la unidad de control seleccionada

5.- Poscondiciones

El usuario ha añadido, modificado o eliminado unidades de control

ANEXO XI. CASO DE USO GESTIONAR PARÁMETROS DE UC

1.- Nombre del Caso de Uso: CU003 Gestionar Parámetros de UC

2.- Descripción

Mediante este caso de uso se podrán adicionar, modificar y/o eliminar los diferentes parámetros asociados a una unidad de control en particular

3.- Precondiciones

1. El usuario se ha autenticado exitosamente.
2. El usuario ha adicionado al menos una unidad de control al sistema.

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el usuario ha accedido a la opción de “Gestionar Parámetros”.
2. El sistema muestra las opciones para adicionar, modificar o eliminar parámetros.
3. El usuario debe seleccionar primero una unidad de control del listado de unidades de control presentes en el sistema.
4. El usuario selecciona la opción de adicionar parámetros (Flujo Alternativo No. 1, Flujo Alternativo No. 2)
5. El sistema le muestra una pantalla para que el usuario introduzca los valores del parámetro.
6. El usuario introduce los valores del parámetro y guarda la información.
7. El caso de uso termina cuando el usuario ha añadido, modificado o eliminado parámetros.

Flujos Alternativos

Flujo Alternativo No. 1

1. Si en el paso 4 el usuario selecciona la opción de modificar parámetros debe seleccionar primero el parámetro de la unidad de control seleccionada anteriormente que desea modificar.
2. El sistema mostrará una pantalla con los valores del parámetro seleccionado para que el usuario realice las modificaciones.
3. El usuario luego de realizar las modificaciones guarda los cambios.

Flujo Alternativo No. 2

1. Si en el paso 4 el usuario selecciona la opción de eliminar parámetros debe seleccionar primero el parámetro de la unidad de control seleccionada anteriormente que desea modificar
2. El sistema pedirá confirmación de la acción a ejecutar.
3. Si el usuario confirma la acción el sistema eliminará el parámetro seleccionado

5.- Poscondiciones

El usuario ha añadido, modificado o eliminado parámetros de la unidad de control seleccionada.

ANEXO XII. CASO DE USO DETECTAR UC

1.- Nombre del Caso de Uso: CU004 Detectar UC

2.- Descripción

Mediante este caso de uso de detecta la presencia de una unidad de control que este instalada

3.- Precondiciones

1. El usuario está autenticado
2. Se está ejecutando el caso de uso CU002 Gestión de UC
3. Se conoce la dirección IP y el puerto asociados a la unidad de control que se desea detectar

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando se ejecuta el caso de uso CU002 Gestión UC
2. El sistema intenta establecer una conexión con la dirección IP y el puerto especificados.
3. Si la conexión se puede establecer satisfactoriamente en un tiempo determinado se considera que la unidad de control está activa y se almacenan los valores de la unidad de control
4. El caso de uso termina cuando se ha recibido respuesta de la solicitud de conexión.

5.- Poscondiciones

Se ha detectado (o no) la presencia de una unidad de control en el sistema

Anexo XIII. Caso de uso Realizar Medición no Atendida

1.- Nombre del Caso de Uso: CU005 Realizar Medición no Atendida

2.- Descripción

Este caso de uso permitirá que se realicen mediciones de los parámetros asociados a las unidades de control sin intervención del operador, sino bajo el control de los mecanismos de planificación del sistema operativo en que corre le aplicación.

3.- Precondiciones

1. Se han introducido unidades de control al sistema.
2. Se han introducido parámetros a las unidades de control.

4.- Flujo de Eventos

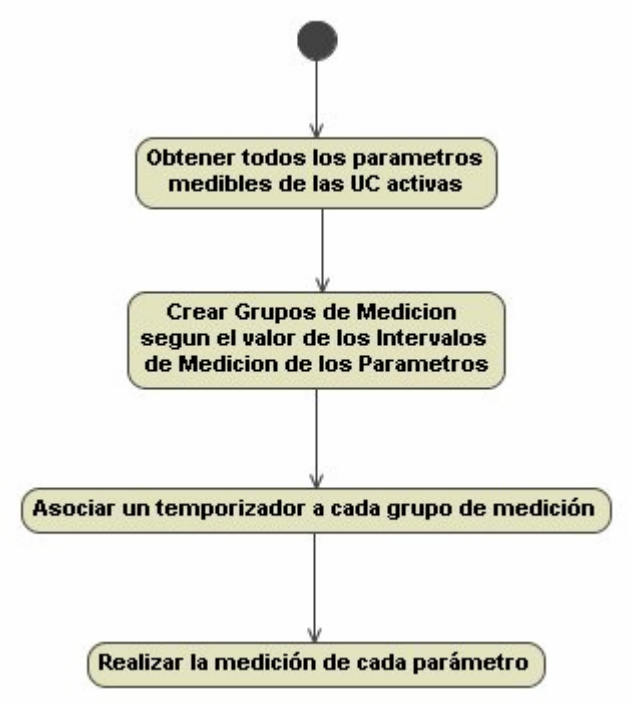
Flujo Básico

1. El caso de uso comienza cuando el sistema planificador del sistema operativo inicia la tarea.
2. El sistema obtiene información acerca de los parámetros que pertenecen a unidades activas en el sistema.
3. El sistema planifica en tiempo la medición de cada parámetro teniendo en cuenta los valores del intervalo de medición asociados a los mismos.
4. El sistema ejecuta la medición de cada parámetro cuando ha llegado el momento.
5. El sistema almacena los valores de las mediciones.
6. Se repite el paso 4 indefinidamente.
7. El caso de uso termina cuando se interrumpe la tarea por un agente externo.

5.- Poscondiciones

Se han realizado mediciones de los parámetros de las unidades de control activas y se encuentran almacenadas.

6.- Diagrama de Actividad



ANEXO XIV. CASO DE USO REALIZAR MEDICIÓN

1.- Nombre del Caso de Uso: CU006 Realizar Medición

2.- Descripción

Mediante este caso de uso el usuario podrá inicial la medición de un parámetro de una unidad de control a voluntad.

3.- Precondiciones

1. El usuario se ha autenticado.
2. Se han añadido unidades de control al sistema.
3. Se han añadido parámetros a las unidades de control.

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el usuario accede a la opción de realizar medición
2. El usuario primero debe seleccionar una unidad de control del listado de unidades de control activas.
3. El usuario debe elegir de la unidad de control seleccionada uno o varios parámetros para realizar sus mediciones.
4. El sistema muestra una pantalla con el resultado de las mediciones de los parámetros seleccionados.
5. El caso de uso termina cuando se ha mostrado la pantalla con el resultado de las mediciones.

5.- Poscondiciones

Se ha realizado y almacenado las mediciones de los parámetros seleccionados.

6.- Diagrama de Actividad

ANEXO XV. CASO DE USO VISUALIZAR ALARMAS EN EL TIEMPO

1.- Nombre del Caso de Uso: CU007 Visualizar Alarmas en el Tiempo

2.- Descripción

Este caso de uso permitirá visualizar las alarmas producidas en un período de tiempos seleccionado por el usuario.

3.- Precondiciones

El usuario se ha autenticado satisfactoriamente

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el usuario accede a la opción de visualizar alarmas en el tiempo
2. El usuario debe seleccionar primero el período de tiempo de interés.
3. El sistema mostrará un listado de las alarmas producidas en ese período de tiempo.
4. El caso de uso termina cuando el sistema ha mostrado el listado de las alarmas producidas en el período de tiempo seleccionado.

5.- Poscondiciones

El usuario habrá obtenido un listado de las alarmas producidas en el período de tiempo seleccionado.

ANEXO XVI. CASO DE USO VISUALIZAR ESTADO UC

1.- Nombre del Caso de Uso: CU008 Visualizar Estado UC

2.- Descripción

Este caso de uso permitirá conocer de forma gráfica el estado de los diferentes parámetros de las unidades de control activas en el sistema.

3.- Precondiciones

El usuario se ha autenticado satisfactoriamente

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el usuario accede a la opción de visualizar el estado de las unidades de control.
2. El sistema mostrará de modo gráfico el estado de los parámetros de las diferentes unidades de control que se encuentren activas en el sistema, en el momento de su ejecución.
3. El caso de uso termina cuando se ha mostrad la pantalla en la que se muestra de modo gráfico el estado de las unidades de control.

5.- Poscondiciones

El usuario ha obtenido de forma gráfica el estado de las unidades de control activas en el sistema

ANEXO XVII. CASO DE USO VISUALIZAR ESTADO PARÁMETROS EN EL TIEMPO

1.- Nombre del Caso de Uso: CU009 Visualizar Estado Parámetros en el Tiempo

2.- Descripción

Mediante este caso de uso el usuario podrá visualizar el estado de los parámetros de una unidad de control en un período de tiempo

3.- Precondiciones

1. El usuario se ha autenticado satisfactoriamente
2. Se han añadido unidades de control al sistema
3. Se han añadido parámetros a las unidades de control.

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el usuario accede a la opción de “Visualizar parámetros en el tiempo”.
2. El usuario selecciona una unidad de control de un listado de unidades activas.
3. El sistema muestra un listado de parámetros de la unidad de control seleccionada.
4. El usuario selecciona uno o varios parámetros del listado.
5. El usuario selecciona el período de tiempo de interés.
6. El usuario selecciona si desea recibir la información de forma gráfica o tabular
7. El sistema muestra la información del estado de los parámetros seleccionados en el período elegido y en el formato indicado.
8. El caso de uso termina

5.- Poscondiciones

El usuario ha visualizado el estado de los parámetros seleccionados en el período elegido y en el formato indicado de la unidad seleccionada.

ANEXO XVIII. CASO DE USO GESTIONAR USUARIOS

1.- Nombre del Caso de Uso: CU010 Gestionar Usuarios

2.- Descripción

A través de este caso de uso el administrador del sistema podrá gestionar los usuarios que tendrán acceso al mismo.

3.- Precondiciones

1. El usuario se ha autenticado satisfactoriamente
2. El usuario es un administrador

4.- Flujo de Eventos

Flujo Básico

1. El caso de uso comienza cuando el administrador accede a la opción de administrar usuarios.
2. El sistema muestra una pantalla con las opciones para insertar, modificar y eliminar usuarios.
3. El administrador selecciona la opción de insertar usuarios.
4. El sistema muestra una pantalla para que el usuario introduzca los datos de los usuarios que accederán al sistema.
5. El administrador introduce los datos necesarios y guarda la información
6. El caso de uso termina cuando el administrador ha añadido, eliminado o modificado la información de un usuario.

Flujos Alternativos

Flujo Alternativo No. 1

1. Si en el paso 3 anterior el administrador decide modificar la información de un usuario debe seleccionar primero uno de un listado de usuarios que le mostrará el sistema.
2. Luego que el administrador seleccione un usuario el sistema mostrará una pantalla con la información del usuario seleccionado.
3. El administrador modifica la información del usuario y guarda los cambios.

Flujo Alternativo No. 2

1. Si en el paso 3 del flujo básico el administrador selecciona eliminar un usuario debe primero seleccionar uno de un listado de usuarios que le mostrará el sistema.

2. El sistema solicita confirmación de la acción a realizarse.
3. Si el usuario confirma la acción el sistema elimina la información del usuario seleccionado.

5.- Poscondiciones

El administrador ha adicionado, eliminado o modificado la información relacionada con un usuario.

Anexo XIX. Caso de Uso Configurar Parámetros Operacionales

1.- Nombre del Caso de Uso: CU011 Configurar Parámetros Operacionales

2.- Descripción

Mediante este caso de uso el usuario podrá configurar los parámetros que regirán el comportamiento del sistema.

3.- Precondiciones

1. El usuario de ha autenticado satisfactoriamente.
2. El usuario es un administrador.

4.- Flujo de Eventos

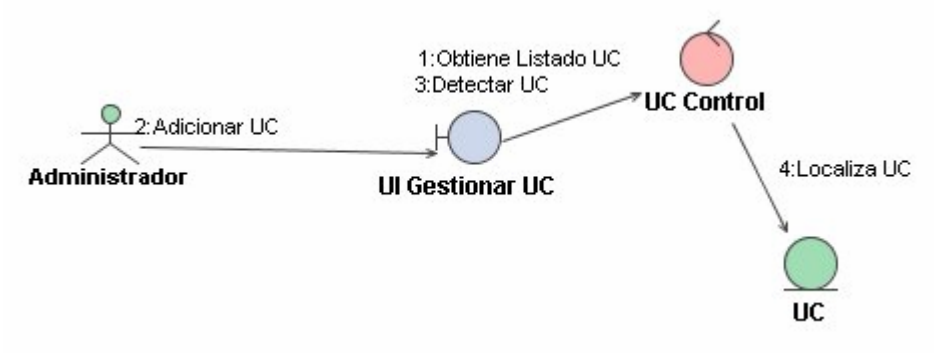
Flujo Básico

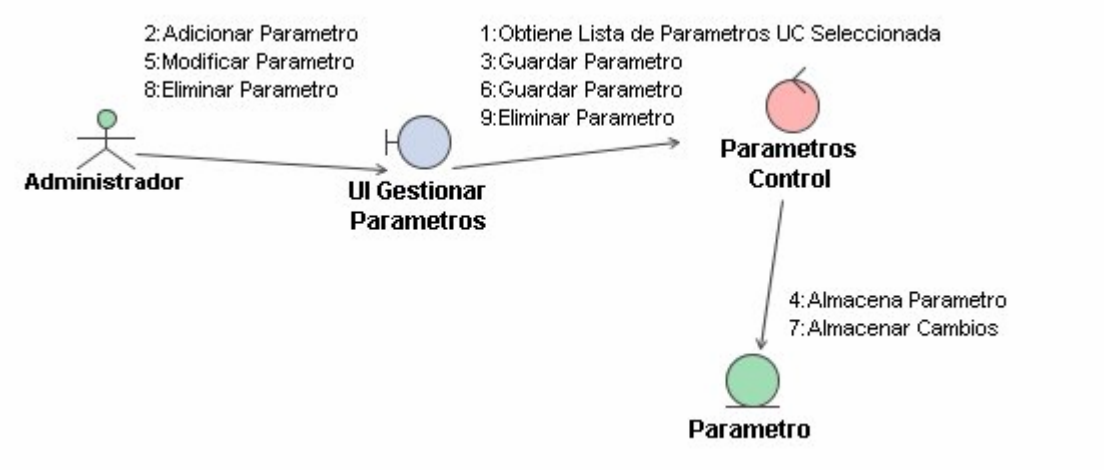
1. El caso de uso comienza cuando el usuario ha accedido a la opción de configurar los parámetros operacionales.
2. El sistema muestra una pantalla para que el usuario modifique los valores de los parámetros operacionales del sistema.
3. El usuario realiza las modificaciones y guarda los cambios.
4. El caso de uso termina.

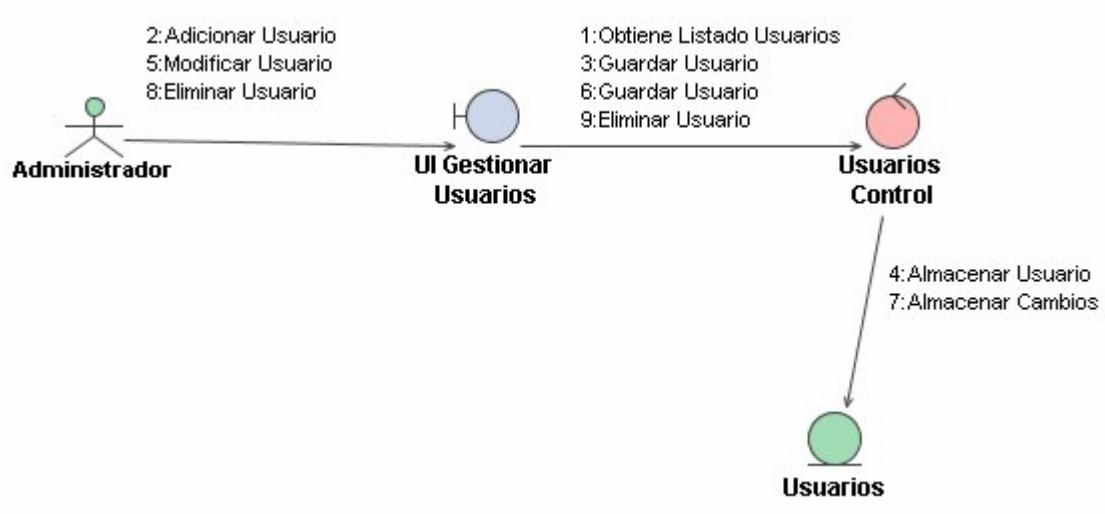
5.- Poscondiciones

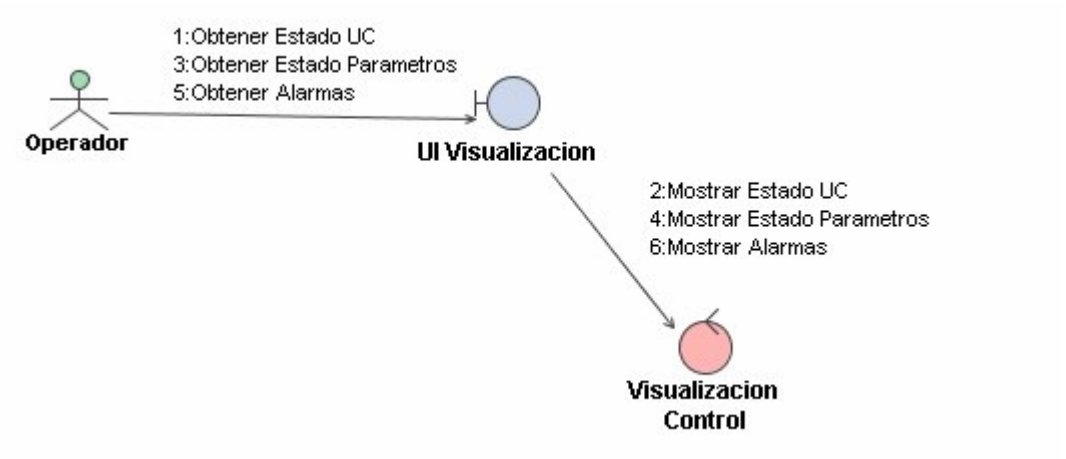
El usuario ha modificado los valores de los parámetros operacionales del sistema

ANEXO XX. DIAGRAMA DE CLASES DE ANÁLISIS

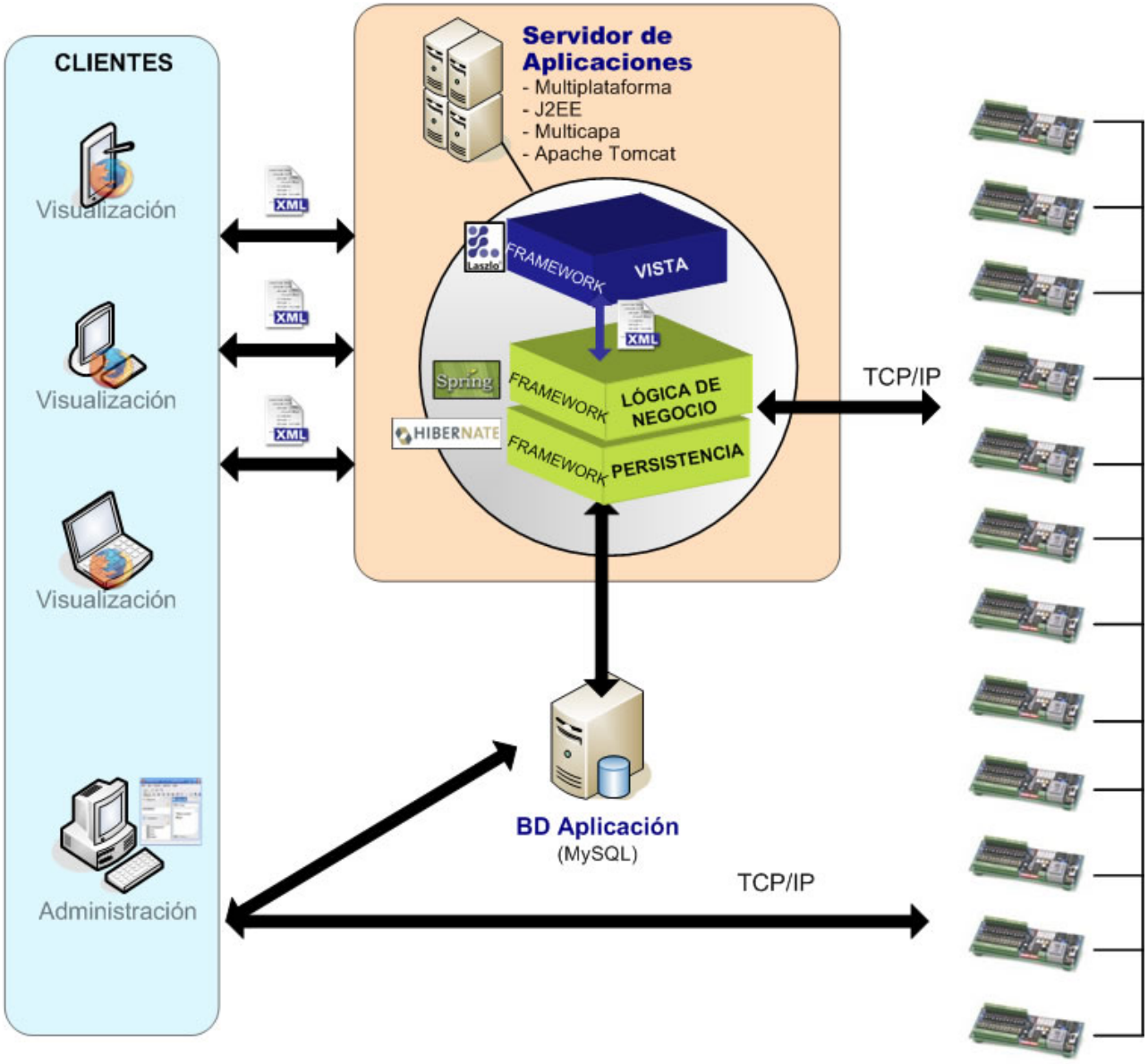


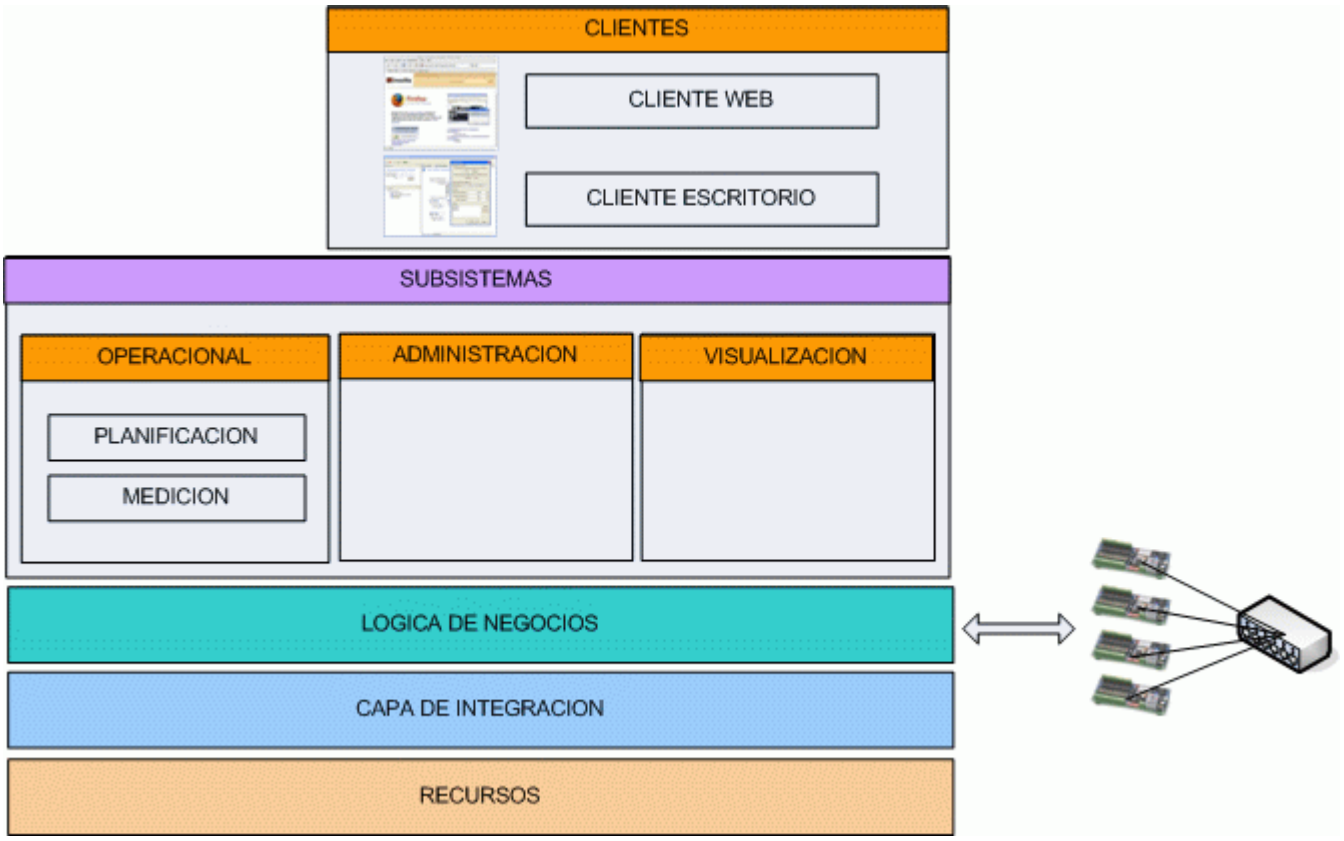




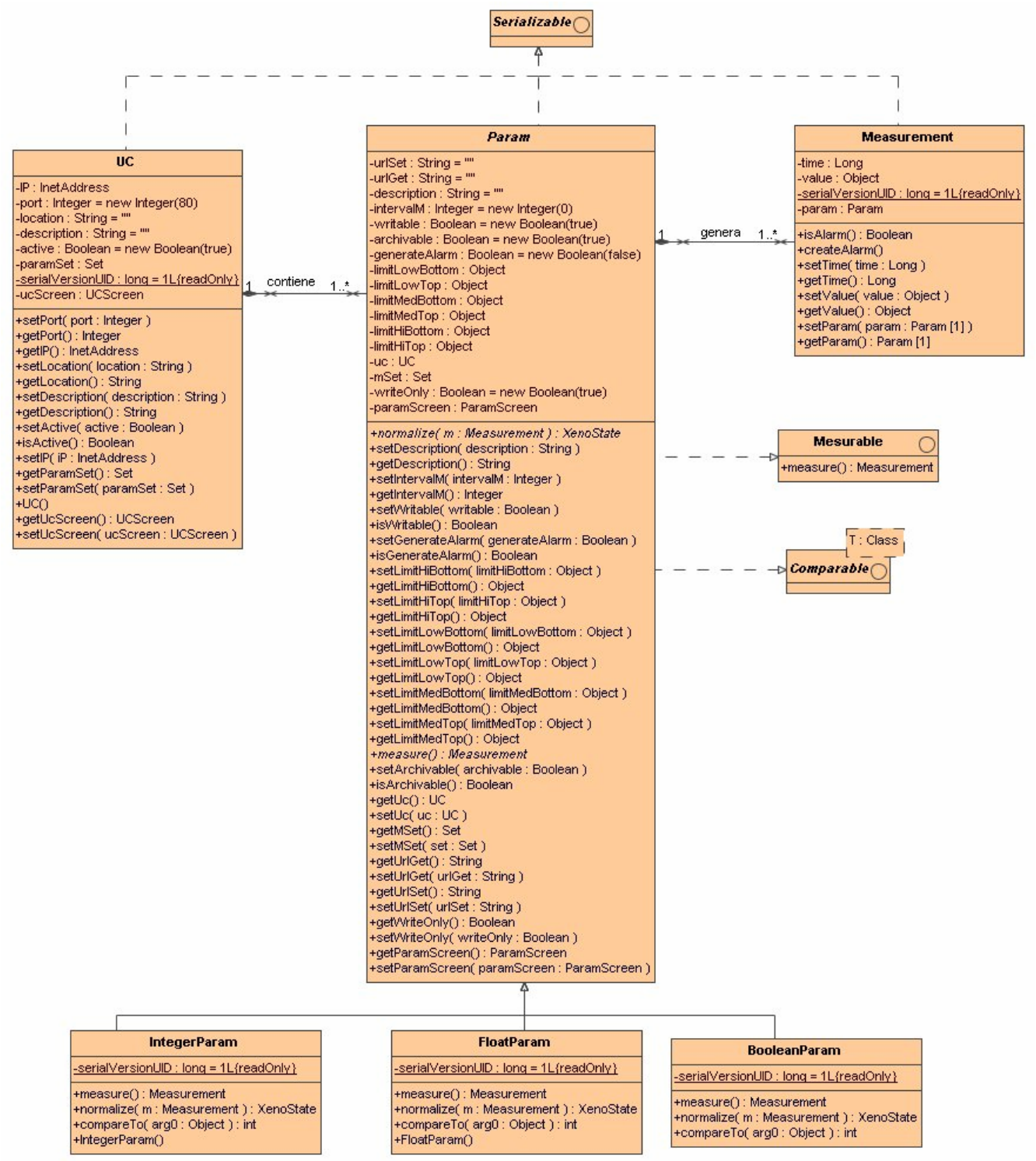

1:Obtener Estado UC
3:Obtener Estado Parametros
5:Obtener Alarmas
Operador
UI Visualizacion
2:Mostrar Estado UC
4:Mostrar Estado Parametros
6:Mostrar Alarmas
Visualizacion
Control

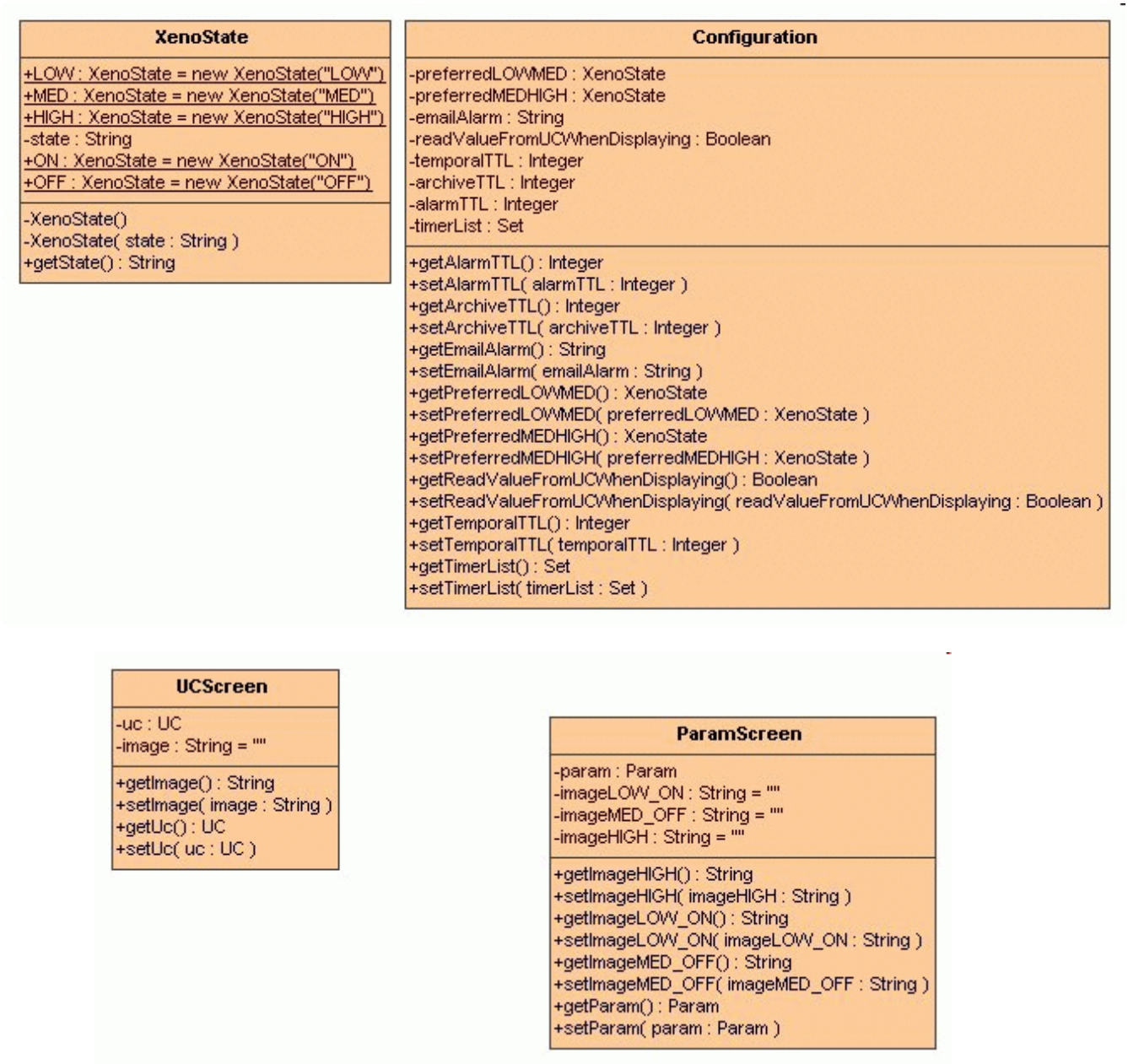
ANEXO XXI. ARQUITECTURA



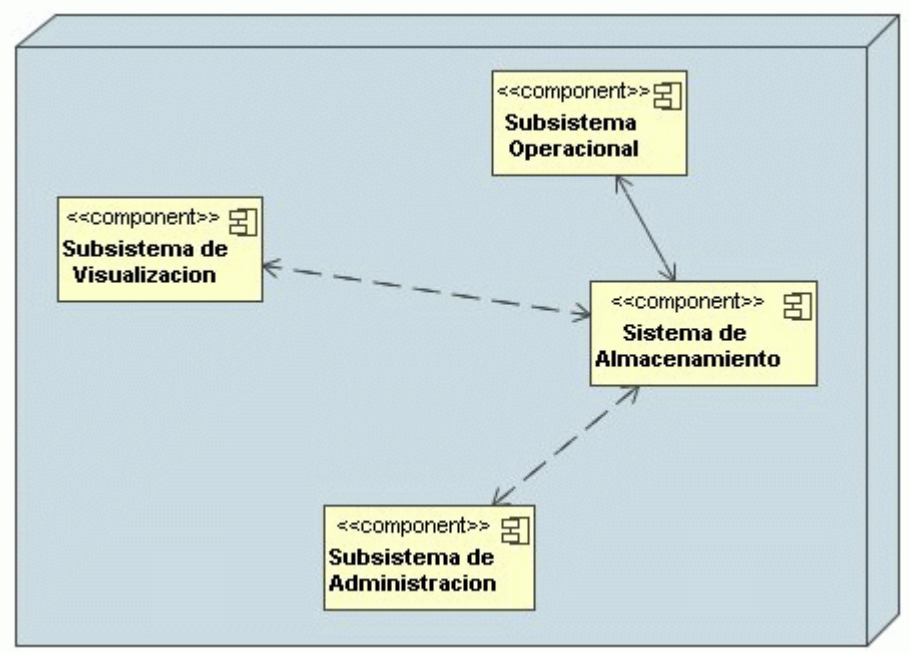

CLIENTES
CLIENTE WEB
CLIENTE ESCRITORIO
SUBSISTEMAS
OPERACIONAL
ADMINISTRACION
VISUALIZACION
PLANIFICACION
MEDICION
LOGICA DE NEGOCIOS
CAPA DE INTEGRACION
RECURSOS

ANEXO XXII DIAGRAMA DE CLASES

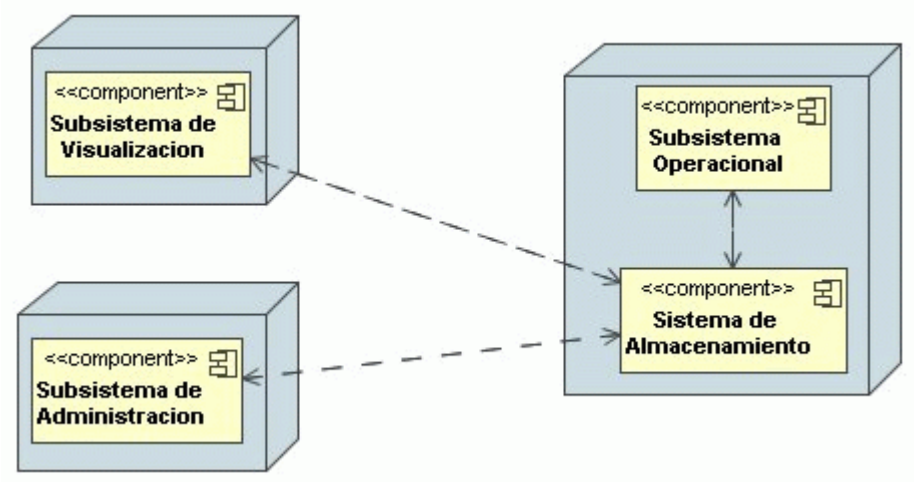



XenoState
+LOW : XenoState = new XenoState("LOW")
+MED : XenoState = new XenoState("MED")
+HIGH : XenoState = new XenoState("HIGH")
-state : String
+ON : XenoState = new XenoState("ON")
+OFF : XenoState = new XenoState("OFF")
-XenoState()
-XenoState(state : String)
+getState() : String
Configuration
-preferredLOWMED : XenoState
-preferredMEDHIGH : XenoState
-emailAlarm : String
-readValueFromUCWhenDisplaying : Boolean
-temporalTTL : Integer
-archiveTTL : Integer
-alarmTTL : Integer
-timerList : Set
+getAlarmTTL() : Integer
+setAlarmTTL(alarmTTL : Integer)
+getArchiveTTL() : Integer
+setArchiveTTL(archiveTTL : Integer)
+getEmailAlarm() : String
+setEmailAlarm(emailAlarm : String)
+getPreferredLOWMED() : XenoState
+setPreferredLOWMED(preferredLOWMED : XenoState)
+getPreferredMEDHIGH() : XenoState
+setPreferredMEDHIGH(preferredMEDHIGH : XenoState)
+getReadValueFromUCWhenDisplaying() : Boolean
+setReadValueFromUCWhenDisplaying(readValueFromUCWhenDisplaying : Boolean)
+getTemporalTTL() : Integer
+setTemporalTTL(temporalTTL : Integer)
+getTimerList() : Set
+setTimerList(timerList : Set)
UCScreen
-uc : UC
-image : String = ""
+getImage() : String
+setImage(image : String)
+getUc() : UC
+setUc(uc : UC)
ParamScreen
-param : Param
-imageLOW_ON : String = ""
-imageMED_OFF : String = ""
-imageHIGH : String = ""
+getImageHIGH() : String
+setImageHIGH(imageHIGH : String)
+getImageLOW_ON() : String
+setImageLOW_ON(imageLOW_ON : String)
+getImageMED_OFF() : String
+setImageMED_OFF(imageMED_OFF : String)
+getParam() : Param
+setParam(param : Param)

ANEXO XXIII. DIAGRAMAS DE DESPLIEGUE



A) Alternativa de Despliegue No. 1



B) Alternativa de despliegue No. 2